3-24-2016

Determination of Fire Control Policies via Approximate Dynamic Programming

Michael T. Davis

Determination of Fire Control Policies via Approximate Dynamic Programming

THESIS

MARCH 2016

Michael T. Davis, Captain, USAF

AFIT-ENS-MS-16-M-100

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENS-MS-16-M-100

DETERMINATION OF FIRE CONTROL POLICIES VIA APPROXIMATE DYNAMIC PROGRAMMING

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

Michael T. Davis, B.S. Mathematics

Captain, USAF

MARCH 2016

AFIT-ENS-MS-16-M-100

DETERMINATION OF FIRE CONTROL POLICIES VIA APPROXIMATE DYNAMIC PROGRAMMING

THESIS

Michael T. Davis, B.S. Mathematics
Captain, USAF

Committee Membership:

Lt Col Matthew J. Robbins, PhD
Co-Advisor

LTC Brian J. Lunday, PhD
Co-Advisor

AFIT-ENS-MS-16-M-100

Abstract

Given the ubiquitous nature of both offensive and defensive missile systems, the catastrophe-causing potential they represent, and the limited resources available to countries for missile defense, optimizing the defensive response to a missile attack is a necessary endeavor. For a single salvo of offensive missiles launched at a set of targets, a missile defense system protecting those targets must decide how many interceptors to fire at each incoming missile. Since such missile engagements often involve the firing of more than one attack salvo, we develop a Markov decision process (MDP) model to examine the optimal fire control policy for the defender. Due to the computational intractability of using exact methods for all but the smallest problem instances, we utilize an approximate dynamic programming (ADP) approach to explore the efficacy of applying approximate methods to the problem. We obtain policy insights by analyzing subsets of the state space that reflect a range of possible defender interceptor inventories. Testing of four scenarios demonstrates that the ADP policy provides high-quality decisions for a majority of the state space, achieving a 7.74% mean optimality gap in the baseline scenario. Moreover, computational effort for the ADP algorithm requires only a few minutes versus 12 hours for the exact dynamic programming algorithm, providing a method to address more complex and realistically-sized instances.

I dedicate this thesis to my beautiful wife and four kids. Without you, I wouldn't have done this.

Acknowledgements

I would like to express my sincere gratitude to my advisors, Lt Col Robbins and LTC Lunday, for their continuous guidance and insight during this effort. Without their support this endeavor would not have been possible.

Michael T. Davis

Table of Contents

List of Figures

List of Tables

DETERMINATION OF FIRE CONTROL POLICIES VIA APPROXIMATE DYNAMIC PROGRAMMING

I. Introduction

Currently, over 30 countries have inventories of theater ballistic missiles [1] while an additional 50 employ multiple launch rocket systems [2]. Both of these weapon systems are capable of causing large amounts of damage and of inflicting a high number casualities on their targets. The proliferation of these weapon systems has increased their destructive potential to a worldwide scale while continued research and development on them has led to the creation of even more capable systems that can be used by their developers to threaten neighboring countries or demand concessions in exchange for halting their production. Even U.S. officials concede that, because of the country's recent focus on counter-terrorism, other world powers have closed the gap on guided munitions technology, and the U.S. is now facing the uncertainty of being able to win a "guided munitions salvo competition" [3].

The threat from these weapons has led to the development and spread of missile defense systems. One of the best known of these systems is Israel's Iron Dome. Developed by Israel and funded mostly by the U.S., the Iron Dome boasts a 90% success rate of destroying incoming rockets headed towards civilian populations, intercepting over 500 rockets during Operation Protective Edge alone according to Israeli officials [4]. The U.S.-developed Patriot system has been in service for over 30 years, seeing use in both Gulf Wars among other conflicts [5], while the system itself has been acquired by 12 other countries [6]. Israel has exported its Iron Dome technology to Canada [7] and continues to work closely with India to develop cutting-edge surface-to-air missiles (SAM) [8]. Still more countries, like Turkey, are seeking to acquire long-range missile defense systems [9], and the U.S. continues to push ahead with missile defense for Europe and Africa [10].

The security these defense systems may provide comes at a significant financial cost. Initial acquisition costs can be billions of dollars depending on the size and scope of the order. For example, the cost to equip Qatar with the Patriot missile defense system in late 2014 was $2.4 billion [11]. Once the system is in place, it must be modernized periodically to counter the evolution of missile threat systems. South Korea paid $770 million for a recent upgrade to its missile defense sytem [12]. Finally, the cost of the interceptor missiles themselves is a large part of the ongoing price of missile defense. The U.S. recently awarded a $1.5 billion contract to Lockheed Martin for an order of its latest interceptors [13] while Saudi Arabia has purchased 600 of the same missiles for $5.4 billion [14].

Given the ubiquitous nature of both offensive and defensive missile systems, the catastrophe-causing potential they represent, and the limited resources available to countries for missile defense, optimizing the defensive response to a missile attack is a valuable endeavor. For a single salvo of offensive missiles launched at a set of targets, a missile defense system protecting those targets must decide how many interceptors to fire at each incoming missile. This decision is the well studied static weapon-target assignment problem. However, missile engagements between an attacker and defender typically extend over many waves of missile launches by the offense. That is, the offense does not launch all of its missiles at once. Instead, it launches subsets of its inventory at selected targets in discrete time periods. Hence, the defense cannot fire all its interceptors at once; it must hold some number of its inventory back in consideration of subsequent attack waves. This component of time is a distinguishing characteristic of the dynamic weapon-target assignment problem.

Previous work by Han et al. [15] provides a framework for the analysis of Integrated Air Defense System (IADS) location placement and subsequent fire control decisions. This paper provides two formulations that address optimization within a multiple-salvo missile engagement setting. Initially, a Markov decision process (MDP) model is developed from the defender standpoint. MDP models are formulated to describe sequential decision-making under uncertainty problems using only the current state of information [16] and are ideally suited to the dynamic weapon-target assignment problem (DWTAP). While the MDP model is inherently a construct for a single decision maker (in this case, the defender), we incorporate a "smart" attacker into the formulation to better inform the resulting optimal policy.

Next, we take an approximate dynamic programming (ADP) approach to deal with the curses of dimensionality, forced on us by larger problem instances. ADP provides many useful strategies and algorithms for solving stochastic optimization problems that are similar to the DWTAP. We apply an approximate policy interation (API) algorithm utlizing least squares temporal differences (LSTD) to solve several DWTAP test instances and then analyze these results as compared to the optimal firing policy.

1.1 Problem Statement

Using the approaches outlined above, we seek a solution to the following problem.

Problem Statement: Given a set of cities C, each with a value $v_i \in \mathbb{R}^+, i \in C$, a set of SAM batteries with corresponding city-covering capabilities, a fixed number of interceptor missiles at each battery, and a fixed number of attacker missiles, we seek the best interceptor allocation policy for a defender.

The remainder of this thesis is organized as follows. Chapter II reviews relevant research on the weapon-target assignment problem which informs the methodologies presented in Chapter III. Chapter III develops the models for the MDP and ADP approaches. Chapter IV provides analytic results of test instances for both formulations. Chapter V concludes the research presented in this thesis and offers suggestions for further study.

II. Literature Review

Overview

The weapon-target assignment problem (WTAP) is a classical operations research problem of great importance to defense-related operations research applications [17]. Simply stated, the WTAP seeks an optimal assignment of some number of weapons to some numbers of targets to maximize the total damage inflicted on the targets [18]. Research focus on the WTAP has only increased through the years as threat systems and platforms proliferate in type and number, to the extent that a weapon-target assignment system that can efficiently solve WTAPs is now a key component of battlefield planning [19].

Cheong [20] lists six factors that are common to any WTAP. They are attacker characteristics, defender characteristics, target characteristics, intelligence available on the opposing force, scenario, and measures of effectiveness of the allocation strategy. Attacker and defender characteristics are composed of the same parameters, albeit for opposing sides. Each side could have only one weapon or missile type or a variety of them. All weapons could perform exactly the same or have higher intercept probabilities depending on the target to which it is assigned. Strategies are specified for both attacker and defender. For instance, missiles could be launched in one large attack or in salvoes that provide the attacker the opportunities to assess success or failure prior to the next launch.

Target characteristics are customizable as well. The chosen target type, such as a point or area target, can simplify or complicate a WTAP considerably. Target value is an important consideration given its role in determining optimality. By manipulating the defense associated with a target entities such as hardened targets are modeled. The information that each side has about the other can range from omniscience to complete ignorance. Depending on the intelligence-derived situational awareness each side has regarding the other, many different combinations of WTAP models are generated. Scenario specifics such as operating in a land- or naval-based environment or modeling unmanned aerial vehicles or ICBMs add to the diversity of the WTAP application. Measures of effectiveness to allow comparison of alternate strategies are chosen based on the problem's parameters, mathematical tractability, or arbitrarily.

Two significant surveys of WTAP literature are Matlin [21] and Eckler and Burr [22]. Matlin reviews the literature based on a set of five submodel characteristics. Each characteristic – the weapon system, the target complex, the engagement, the damage model, and the algorithm – is partitioned alphabetically based on the complexity of the assumptions within that particular characteristic. Matlin primarily focuses on asset-based, offensive allocation models, both with single and multiple missile types, but also includes models that consider the defender's allocation of interceptors. Eckler and Burr provide an extensive and thorough look at even wider range of models. Besides allocation models, the authors examine many more technical aspects and variations of the target-based WTAP model.

Work on the WTAP is traced back to the 1950s and 1960s. Manne [23] developed a linear programming approximation, while Bradford [24] and Day [25] studied WTAP modeling issues including its decomposition into subproblems with subsequent reconstitution. In 1975, Croucher [26] applied game theory to a small antiballistic missile defense scenario in which the attacker targets 3 defensive assets using a ballistic missile which contains 28 multiple reentry vehicles. The defense is allotted 16 weapons for assignment, and each of its assets is assigned a different value. After first establishing the existence of pure strategies, the instance is solved by finding the min-max and max-min solutions.

Exact algorithms for some WTAP formulations are proposed in the literature.

The most well known is found in den Broeder et al. [27], which presents the minimum marginal return (MMR) algorithm to solve the case of identical weapons. However, the general WTAP is NP-complete, as proven via a reduction from the exact cover problem by Lloyd and Witsenhausen [28].

The two fundamental classes of the WTAP are the static and dynamic WTAPs. According to Xin [29], in a static WTAP, all parameters for the problem are known, and all weapons are assigned to the targets in a single stage. Comparatively, in a dynamic WTAP, multiple stages require weapon-target assignments with each of these stages subsequently evaluated to make future assignment decisions to obtain the best global assignment.

2.1 Static WTAP

Zeng et al. [30] solves the static WTAP using discrete particle swarm optimization. The approach leverages advantages from both genetic algorithms and particle swarm optimization to effect a solution. The authors utilize a mutation strategy, much like those implemented in genetic algorithms, to prevent the procedure from becoming trapped in a local optimum. They also discretize the particle swarm optimization model to apply it towards the WTAP. For a 60-weapon, 60-target test instance the proposed algorithm converges in the same amount of time to a better solution than either a standard genetic algorithim or a genetic algorithm with a greedy eugenic.

A genetic algorithm approach by Lee et al. [19] to the static WTAP incorporates a novel method of gene recombination. The authors express the weapon-target assignment as a chromosome where the position of the gene represents the weapon assigned and the value of the gene represents the target. Based upon the value of the target and the probability of the assigned weapon killing the target, the authors label a gene as "good" if it has the highest such combination of value and kill probability. The algorithm keeps common "good" genes between two parents and uses them to produce the next generation of solutions. The authors also propose a "greedy eugenic" which effectively intensifies the local search for the best solution. The algorithm solves a 120-weapon, 80-target instance with good results.

Madni and Andrecut [18] present simulated annealing and threshold accepting approaches to the static WTAP. The authors test them against the MMR algorithm to determine the solution quality. Both of the proposed heuristics solve static WTAP instances of up to 200 weapons and 200 targets close to optimality in a matter of seconds.

Wacholder [31] considers a neural network-based approach to solve the static WTAP. The formulation in this research considers a nonlinear combinatorial optimization problem that restricts each weapon platform to assigning only one interceptor per target. In the neural network representation, Wacholder defines weapon assignment variables as the output signals of the neurons and defines the objective function (i.e., the total expected value of missed targets) and constraints by energy functions. The developed algorithm produces useful solutions for real-time implementation in combat situations while the structure of the formulation also allows peacetime analysis of various parameter settings to design an optimal defensive posture.

Ahuja et al. [17] exploit the special structure of the static WTAP to formulate linear programming, mixed integer programming, network flow, and combinatorial lower-bounding schemes for proposed algorithms. The authors formulate the standard nonlinear WTAP as an integer programming problem with a convex objective function value. They view this formulation as a generalized network flow problem with convex costs which they approximate by a piecewise-linear convex function so that the modified problem solution gives a lower bound to the general problem. To obtain the linear programming-based lower-bounding scheme, Ahuja et al. relax the integrality constraints of the weapon assignment variables. To obtain the mixed integer-based lower-bounding scheme, the authors maintain the integrality of the weapon assignment variables while transforming the piecewise-linear convex functions to linear cost functions.

The authors develop a minimum cost flow-based lower-bounding scheme by interpreting the WTAP objective function as maximizing the expected damage to the targets as opposed to minimizing the survival value of the targets. Using that interpretation, they develop an upper bound by formulating the WTAP as a maximum cost flow problem. Subtracting the upper bound from the total value of the targets provides the lower bound.

The authors develop a maximum marginal return-based lower-bounding scheme by underestimating the survival of a target when hit by a weapon. Instead of allowing any weapon to hit the target, the authors assume that the best weapon hits the target. This assumption limits the WTAP to only one weapon type, the best, making any solution to this formulation a lower-bound on the case including weapon types of inferior capability.

The authors propose several algorithms for solving the WTAP. They develop and implement a branch-and-bound algorithm using three of the four lower-bounding schemes outlined above. They exclude the linear-programming scheme due to its inability to generate tight bounds. For smaller instances, a breadth-first strategy produces the best results while for larger instances, the depth-first strategy produces the best. The proposed branch-and-bound algorithm is the first exact algorithm able to solve up to an 80-weapon, 160-target instance of the WTAP in moderately good time.

Ahuja et al. also propose a very large-scale neighborhood (VLSN) search algorithm. Prior to implementing the VLSN algorithm, they employ a construction heuristic that solves a sequence of minimum cost flow formulations of the WTAP and provides a good starting feasible solution. The proposed WTAP VLSN employs cyclic and path multiexchanges to create neighbors. The multiexchanges allow weapons to be reassigned from target to target. The algorithm finds mostly two-exchanges, though searches for up to five-exchanges. For an 80-weapon WTAP instance, this cyclic exchange search equates to a neighborhood size of about 3 billion solutions. The authors develop an implicit enumeration algorithm by leveraging the idea of improvement graphs to accommodate the extreme neighborhood size.

Of the three lower-bounding schemes tested, the mixed integer programming scheme yields the tightest bounds but produces the longest running times. The minimum cost flow-based method gives very tight lower bounds when the number of weapons is less than the number of targets, whereas the maximum marginal return-based lower-bounding scheme is computationally efficient but produces not as tight bounds.

The authors employ their branch-and-bound algorithm with each of the lower-bounding schemes. For certain instances and schemes, the authors obtain no solutions within 48 hours running time using the minimum cost flow-based and maximum marginal return-based lower bounds. However, the branch-and-bound algorithm, when using the mixed integer programming lower-bounding scheme, produces consistent results in a timely fashion.

While meant to find starting solutions for the VLSN, the construction heuristic performs extremely well and obtains optimal solutions for over half the test cases. For instances where the heuristic could not reach optimality, the VLSN either finds the optimal solution or comes within less than 0.01%. The heuristic and VLSN solves within 3 seconds test instances of up to 200 weapons and 400 targets, the largest instance found in the literature.

2.2 Dynamic WTAP

According to Murphey [32], the dynamic WTAP is formulated in one of two manners. The first model assumes that all targets are known from the start, while the second assumes that only a subset are known while others may be revealed stochastically. The first formulation is also known as a shoot-look-shoot model and is the more widely studied. The second formulation allows for additional targets to become known as time progresses, making the decision problem one of how many weapons to assign to the known targets and how many to reserve for future targets that may present themselves. This formulation is a stochastic demand problem and is introduced and studied in Murphey [33]. Ahner and Parson [34] exploit the structure of the stochastic demand problem to optimally solve a two-stage formulation.

Hosein and Athans [35] present the general dynamic WTAP formulation. Due to the complexity of the problem, the authors limit the number of stages to two, while assuming that weapon-target kill probabilities depend only the asset at which the target is aimed. They develop an upper bound for the optimal value as well as a heuristic solution method.

As is the case for the static WTAP, heuristic methods are the most common approach to solving the dynamic WTAP given its computational complexity. Xin et al. [36] use three rules based on the potential damage of an incoming missile and potential benefit of a particular interceptor assignment to develop a heuristic that solves the asset-based, dynamic WTAP. Both Wu [37] and Khosla [38] employ a genetic algorithm (GA) approach to the dynamic WTAP. Wu's modified GA approach allows for the dynamic allocation of weapons to new targets without restarting the algorithm, thus giving the algorithm more time to find higher quality weapon-target pairings. Khosla combines a GA with a simulated annealing heuristic to optimize the weapon-target assignment when faced with resource and timing constraints.

Lötter and van Vuuren [39] examine four classes of the WTAP to provide support to the Threat Evaluation and Weapon Assigment Decision Support System (TEWA DSS). In particular, the authors focus on the optimization of the Weapon Assignment subsystem which is responsible for providing the fire control officer (FCO) with high-quality assignments of surface-based weapons to airborne threats. The examined classes, listed in increasing levels of complexity, are the single-objective, static WTAP; the multi-objective, static WTAP; the single-objective, dynamic WTAP; and the multi-objective, dynamic WTAP. Assuming models for all WTAP classes are incorporated into the TEWA DSS, the authors propose that the FCO utilize a decision tree in the predeployment stage to configure the Weapon Assignment subsystem.

The authors solve a realistic missile defense scenario using the first three classes presented. The multi-objective, dynamic class is not modeled. For the static models, they implement two different genetic algorithms. For the single-objective, dynamic model, the authors solve with a simulated annealing heuristic. The approach for all three formulations provides high quality, quickly attained solutions for the FCO to choose from.

Defended assets are typically of more than a single type or purpose. Hosein et al. [40] intersperses command, control, and communication (C3) nodes among the typical defended assets in their model. The authors study this structure for both dynamic and static WTAPs. Destruction of the C3 system renders the subset of interceptors under its management useless and increases the vulnerability of the defended assets. Attacking the C3 nodes is not without risk for the offense as attempts at these sites leave fewer missile available for targeting the remaining assets. To demonstrate how the complexity of the WTAP with C3 can grow, the authors impose kill probability and asset value constraints. They then relax the constraints and increase the problem instance size. The authors show that model formulation as a dynamic versus static WTAP significantly increases the effectiveness of the defense.

Soland [41] uses stochastic dynamic programing to solve an asset-based, dynamic WTAP. The model formulation is limited to one asset as well as weapon-target kill probabilities that are the same within each stage. The author provides numerical results as well as some extensions regarding the number of interceptors remaining for the defense.

Most dynamic WTAP formulations assume that the defender accurately predicts the asset that an offensive missile is targeting. Leboucher et al. [42] relax this assumption so that a defender can only tell the particular region that is being targeted; the region may have one or more assets needing defense. The authors propose a combined evolutionary game theory and discrete particle swarm optimization approach to solve the problem, and they provide computional results using numerical simulation.

Karasakal [43] develops an integer linear programming model that also addresses the defensive effectiveness of a naval task group. The formulation assumes a shoot-look-shoot policy and considers both point defenses as well as area defenses. Point defenses intercept only missiles fired directly at the ship they are onboard whereas an area defense fires at all targets within its range. From the standard nonlinear integer programming model, Karaskal creates two linear integer programming models using a logarithmic linearization process. The first model does not guarantee an optimal assignment solution; it minimizes the total deviation from a desired probability of not having any attacking missiles leak through the defense. The second model also does not seek the best weapon-target assignment. Instead, it minimizes the maximum deviation from the desired probability.

The author randomly generates 36 test instances ranging from 3 to 81 anti-ship missiles and surface-to-air missiles to test both proposed models. The first model produces the better results in 22 out of 36 instances whereas the second model performed better in six cases. Karaskal performs ten runs each for five of the previous instances with each model taking less than a second on average to provide an approximate weapon-target solution.

Bertsekas et al. [44] model a dynamic WTAP that is considerably more complex than the static WTAP. Instead of making one assignment of weapons to targets, the defense must decide how many weapons to employ against the current attack and how many to keep in reserve for subsequent attacks. Since Bellman's "curse of dimensionality" denies the possibility of an exact solution for problems of even moderate size, the authors apply a class of reinforcement learning methods called neuro-dynamic programming to deal with the dimensionality of the dynamic WTAP.

The neuro-dynamic programming framework for the dynamic WTAP employs suboptimal solution methods to approximate the optimal function via neural networks and simulation. The authors develop four approximate policy iteration methods that generate a sequence of policies that allow for the approximate evaluation of the optimal functions. The approximate policy iteration methods require the potential solutions to undergo a follow-on screening process to effectively select the best one. The authors instantiate 23 test cases using three asset types, one missile type, and one interceptor type, with the number of interceptors and missiles ranging from 40 to 60. Results show that none of the methods dominate the others with tradeoffs existing among the various policy iterations. Tuning of particular aspects of each is likely to lead to more efficient solution processes.

III. Methodology

Joint Publication 3-01 (JP3-01) [45] recognizes two main enemy threats to an integrated air defense system: air threats (i.e., fighters and bombers) and ballistic missile (BM) threats. Of the two, BMs are considered more difficult to counter by offensive targeting since, in general, they have smaller logistical footprints and are more easily maneuvered and concealed. Since an enemy's BM assets are unlikely to be completely destroyed prior to launch, it is essential to devise a defensive strategy to counter their use. JP3-01 assumes an IADS's ability to identify and target incoming BMs to include impact points, and it outlines planning considerations for countering a BM salvo against a set of defended assets. These considerations – placement of SAM sites, return salvo size, interceptor inventories, and firing doctrine – attempt to enable the best possible response to an attack salvo. Since it is reasonable to assume that an attacker has a limited supply of BMs and a limited number of launchers, it is also reasonable to assume that an enemy would choose to stage an attack over several salvos to enable efficient use of limited assets via iterative battle damage assessment of the defended assets and to allow for reloading and/or repositioning of launchers. Thus, we view an enemy BM campaign as a series of "look-shoot" engagements.

In our formulation, the defender has a set of cities, each having a value, it wishes to protect from incoming missiles using a predetermined configuration of SAM sites with preallocated supplies of interceptors that cannot be replenished. SAM sites are assumed to be collocated with a city (though not every city may have one) and to have a predefined protection radius of cities each SAM site could defend. Cities, but not SAM sites, are assumed to be destroyed if at least one attacking missile targeting the city is not successfully intercepted. Implied herein is an attacker strategy that is counter-value focused rather than counter-force focused. The attacker has some finite number of missiles available for carrying out attacks which may be launched in

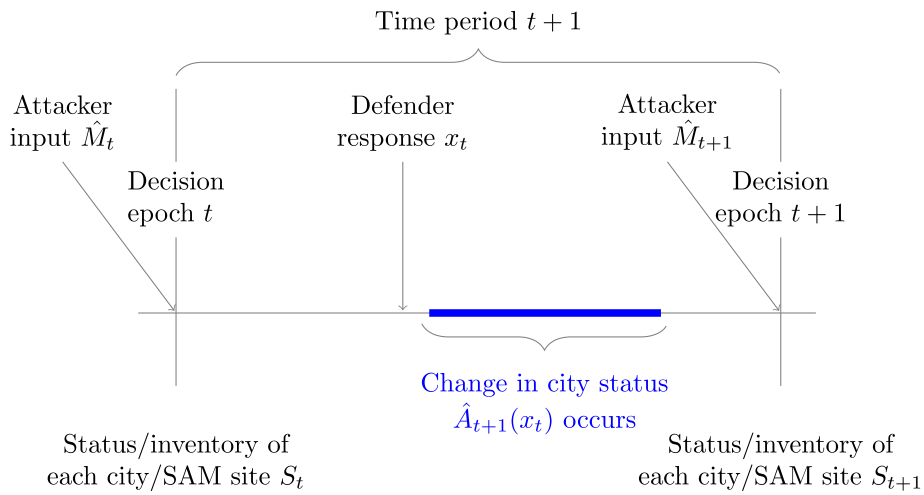


Figure 1. Diagram outlining the timing of events for the "look-shoot" MDP model.

multiple salvos. The attacker can observe the status of each city prior to launching an attack. Once an attack is launched, the defense can identify which city has been targeted by each missile. The defense must then decide how many interceptors to allocate from among its SAM sites to each incoming missile and how many to keep in reserve for repelling subsequent BM attacks. We wish to maximize the expected value of the cities that remain after all attack salvos have been launched. Equivalently, we wish to minimize the expected total cost of destroyed cities over all decision epochs. Figure 1 shows a timing diagram of the model.

3.1 MDP Model

The Markov Decision Process (MDP) model is formulated in the following manner.

1. Let $\mathcal{T} = \{1, 2, ..., T\}, T \leq \infty$ be the set of decision epochs.

2. The state space consists of three components: the status of each city, the inventory of each SAM site, and the attack "vector".

   (a) The city status component is defined as

$$A_t = (A_{ti})_{i \in \mathcal{A}} \equiv (A_{t1}, A_{t2}, ..., A_{t|\mathcal{A}|}),$$

where $\mathcal{A} = \{1, 2, ..., |\mathcal{A}|\}$ is the set of all cities, and $A_{ti} \in \{0, 1\}$. A_{ti} is the status of city $i \in \mathcal{A}$ at decision epoch t with one indicating the city is alive and zero indicating the city is destroyed.

(b) The SAM inventory status is defined as

$$R_t = (R_{ti})_{i \in \mathcal{A}} \equiv (R_{t1}, R_{t2}, ..., R_{t|\mathcal{A}|}),$$

where $R_{ti} \in \{0, 1, ..., r_i\}$ and r_i = initial inventory of interceptors at SAM site $i \in \mathcal{A}$. R_{ti} is the number of interceptors at SAM site $i \in \mathcal{A}$ at decision epoch t.

(c) Let $\hat{\mathcal{M}}_t = \{1, 2, ..., |\hat{\mathcal{M}}_t|\}$ be the set of all fired attacker missiles at decision epoch t. $\hat{\mathcal{M}}_t$ is the collection of observed incoming BMs that must be targeted by the defense at time t. The attack "vector" is defined as

$$\hat{M}_t = (\hat{\mathcal{M}}_{ti})_{i \in \mathcal{A}} \equiv (\hat{\mathcal{M}}_{t1}, \hat{\mathcal{M}}_{t2}, ..., \hat{\mathcal{M}}_{t|\mathcal{A}|}),$$

where $\hat{\mathcal{M}}_{ti} \subseteq \hat{\mathcal{M}}_t$ is the set of missiles fired at city i at decision epoch t, and the tuple $\hat{M}_t$ forms a disjoint set partition of $\hat{\mathcal{M}}_t$. The information provided by $\hat{M}_t$ is available to the defender at time t. However, the arrival of new information, $\hat{M}_{t+1}$, is random and could be conditioned on A_{t+1}. Let $\mathbb{P}^{\hat{M}_t}(m) = \mathbb{P}(\hat{M}_t = m | A_t)$ denote the probability distribution of the attacker BM salvo $\hat{M}_t$. This distribution is conditioned on A_t meaning that the battle damage assessment capabilities of an attacker will determine the likelihood that a particular attack vector arrives to the system.

Using these components, we define $S_t = (A_t, R_t, \hat{M}_t) \in \mathcal{S}$ as the state of the system at decision epoch t, where $\mathcal{S}$ is the set of all possible states.

3. At each epoch t, the defender must decide how many interceptors to assign to

each missile targeting a city. The defender must make this choice from among the SAM sites that have the given city within their respective protection radii. From the *a priori* placement of SAM sites relative to the cities, we can deduce a coverage matrix for the entire defended area. From this coverage matrix, we can determine which SAM sites can intercept each incoming missile. Let $x_{tij} \in \mathbb{N}^0$ be the number of interceptors fired by SAM site $i \in \mathcal{A}$ against missile $j \in \hat{\mathcal{M}}^A_{ti}$ at decision epoch t, where $\hat{\mathcal{M}}^A_{ti}$ is defined as the set of missiles that can be intercepted by SAM site i at decision epoch t. Let $x_t = (x_{tij})_{i \in \mathcal{A}, j \in \hat{\mathcal{M}}^A_{ti}}$ denote our decision vector. We define the set of all feasible defender actions (i.e., assignment of interceptors to missiles) as

$$\mathcal{X}_{S_t} = \{x_t : \sum_{j \in \hat{\mathcal{M}}^A_{ti}} x_{tij} \leq R_{ti}, \ \forall \ i \in \mathcal{A}\},$$

where the constraint $\sum_{j \in \hat{\mathcal{M}}^A_{ti}} x_{tij} \leq R_{ti}$ ensures that each SAM site $i \in \mathcal{A}$ cannot fire more interceptors than it has in inventory.

4. We define transition functions and transition probability functions to characterize how the system evolves from one state to another as a result of decisions and information [46]. The state transition function is defined as $S_{t+1} = S^M(S_t, x_t, W_{t+1})$, where $W_{t+1} = (\hat{A}_{t+1}, \hat{M}_{t+1})$. W_{t+1} represents all the information (i.e., city status and attacker BM salvo) that becomes known at decision epoch $t+1$. We define the city status transition function as

$$A_{t+1,i} = \begin{cases} 0 & \text{if } A_{ti} = 0, \\ \hat{A}_{t+1,i}(x_t) & \text{otherwise}, \end{cases} \quad \forall i \in \mathcal{A},$$

where $\hat{A}_{t+1,i}(x_t)$ is a random variable representing the status of city i after salvo $\hat{M}_t$ and the interceptor allocation decision x_t. This information depends on x_t since the number of interceptors fired at the inbound BMs affects a city's probability of survival. We define the inventory status transition function as

$$R_{t+1,i} = R_{ti} - \sum_{j \in \hat{\mathcal{M}}_{ti}^{A}} x_{tij}, \ \forall \ i \in \mathcal{A},$$

and note that the city status transition function is stochastic whereas the inventory status transition function is deterministic.

The probability of transitioning from state S_t to S_{t+1} is conditioned on both the state of the system and the action chosen by the defender at decision epoch t. We assume the defender has one interceptor type, the attacker has one missile type, and that any missile that is unintercepted results in the certain destruction of the targeted city. We define $q \in (0,1)$ to be the probability an attacking missile survives being targeted by a single interceptor. Then $\rho_{tj} = \prod_{i \in \mathcal{A}} q^{x_{tij}}$ is the probability that missile $j \in \hat{\mathcal{M}}_t$ survives being targeted by all interceptors fired against it at decision epoch t. We define

$$\psi_{ti} = \begin{cases} \prod_{j \in \hat{\mathcal{M}}_{ti}} (1 - \rho_{tj}) & \text{if } \hat{\mathcal{M}}_{ti} \neq \emptyset, \\ 1 & \text{if } \hat{\mathcal{M}}_{ti} = \emptyset, \end{cases}$$

as the probability that city $i \in \mathcal{A}$ survives to decision epoch $t+1$. Thus $\hat{A}_{t+1,i}(x_t)$ follows a Bernoulli probability distribution with parameter ψ_{ti}. Then

$$p_t(S_{t+1}|S_t, x_t) = \begin{cases} \mathbb{P}^{\hat{M}_{t+1}}(m) \prod_{i \in \mathcal{A}} \psi_{ti}^{A_{t+1,i}} (1 - \psi_{ti})^{A_{ti} - A_{t+1,i}} & \text{if } A_{ti} \geq A_{t+1,i} \text{ and} \\ & R_{t+1,i} = R_{ti} - \sum_{j \in \hat{\mathcal{M}}_{ti}^{A}} x_{tij}, \\ & \text{and } \hat{M}_{t+1} = m, \\ 0 & \text{otherwise,} \end{cases}$$

is the transition probability function from state S_t to S_{t+1}.

5. At each decision epoch t, the defender incurs an expected cost as a result of its decision. We define this cost as $\hat{C}(S_t, x_t, \hat{A}_{t+1,i}) = \sum_{i \in \mathcal{A}} v_i(A_t - \hat{A}_{t+1,i})$, where v_i is the value of city $i \in \mathcal{A}$. We rewrite the cost function in terms of only the current state and decision by taking its expected value

$$C(S_t, x_t) = \mathbb{E}\left\{\sum_{i \in \mathcal{A}} v_i(A_t - \hat{A}_{t+1,i})|S_t, x_t\right\}.$$

To determine the optimal policy, we must find a solution to the Bellman equations

$$J_t(S_t) = \min_{x_t \in \mathcal{X}_{S_t}} (C(S_t, x_t) + \gamma \mathbb{E}\{J_{t+1}(S_{t+1})|S_t, x_t\}). \tag{1}$$

These equations are alternately referred to as optimality equations or value functions. The parameter $\gamma \in (0, 1)$ is a discount factor that represents our expectation of BM campaign length, or the expected number of decision epochs T. We note the following relationship

$$\mathbb{E}[T] = \frac{1}{1-\gamma}. \tag{2}$$

The number of attack salvos for a BM campaign is controlled by the attacker and not dependent on the defender's actions. Incorporating an uncertain horizon allows us to model the randomness of how long a missile engagement may last.

6. Using the reward function, we define the decision function as

$$X_t^\pi(S_t) = \underset{x_t \in \mathcal{X}_{S_t}}{\operatorname{argmin}}(C(S_t, x_t) + \gamma \mathbb{E}\{J_{t+1}(S_{t+1})|S_t, x_t\}),$$

where π represents a policy, or specified defender actions for each state.

We wish to determine the policy π^* that minimizes the expected total discounted cost of destroyed cities over all epochs. Thus our objective is

$$\min_{\pi \in \Pi} \mathbb{E}^{\pi} \left\{ \sum_{t=0}^{T} \gamma^t C(S_t, X_t^{\pi}(S_t)) \right\}.$$

The notation $\mathbb{E}^{\pi}$ denotes that the expectation depends on the defender's chosen policy.

3.2 ADP Model

The MDP model formulation provides an elegant framework for the interceptor allocation problem. However, the application of exact dynamic programming algorithms to the problem is limited to very small instances. This limitation exists because our problem suffers from the curses of dimensionality. For example, consider the dimensionality of the state space $\mathcal{S}$, where $S_t = (A_t, R_t, \hat{M}_t) \in \mathcal{S}$ is an arbitrary state. The tuples A_t, R_t and $\hat{M}_t$ represent the status of each city, the status of each SAM battery's inventory, and the attack vector at decision epoch t, respectively. Since city status is binary there are $2^{|\mathcal{A}|}$ possibilities for A_t. Since SAM batteries are collocated at cities, there are $\prod_{i \in \mathcal{A}} (r_i + 1)$ possibilities for R_t. Let M be the maximum number of attacker missiles that can be fired across all cities at any epoch t. That is, the attacker may fire up to and including M missiles total at each decision epoch. Then there are $\binom{|A|+M}{M}$ possibilities for $\hat{M}_t$. Hence there are $2^{|\mathcal{A}|} \cdot \prod_{i \in \mathcal{A}} (r_i + 1) \cdot \binom{|A|+M}{M}$ possibilities for S_t. This means that given a problem instance of 10 cities and 5 SAM batteries with 10 interceptors each and an attacker firing up to 10 missiles, we would have 3×10^{13} possible states. Because classical dynamic programming algorithms such as policy iteration and value iteration that solve the Bellman equations exactly rely on enumeration of the state space, all but the smallest problem instances are computationally intractable.

Approximate dynamic programming (ADP), or computational stochastic optimization, provides an alternative set of solution strategies that can be applied to problems that suffer from one or more curses of dimensionality. A first key strategy of ADP is that of stepping forward in time, in contrast with recursively solving the Bellman equations, a standard technique. By stepping forward, we can no longer solve the Bellman equations, which eliminates the need for enumeration of the state space. Instead of applying backward induction, we simulate the stochastic process forward, generating samples of possible outcomes and approximating how we make decisions.

Although stepping forward allows us to handle large state spaces, there remain other challenges to contend with such as approximating the expectation. A second key idea of ADP – the construct of a *post-decision state* variable – allows us to avoid this step. Van Roy *et al.* [47] are the first to use this term, while Powell and Van Roy [48] define the post-decision state variable as the state at time t immediately after making a decision x_t but prior to the arrival of any new information $\hat{W}_{t+1}$. The general state transition function $S_{t+1} = S^M(S_t, x_t, W_{t+1})$ can be broken into two steps

$$S_t^x = S^{M,x}(S_t, x_t),$$

and

$$S_{t+1} = S^{M,W}(S_t^x, W_{t+1}),$$

where S_t^x is the post-decision state variable. For our problem, the post-decision state is given by $S_t^x = (A_t^x, R_t^x)$, where $A_t^x = (A_{ti}^x)_{i\in\mathcal{A}}$ and $R_t^x = (R_{ti}^x)_{i\in\mathcal{A}}$. Let

$$A_{ti}^x = \psi_{ti}$$

and

$$R_{ti}^x = R_{ti} - \sum_{j\in\hat{\mathcal{M}}_{ti}^{\mathcal{A}}} x_{tij}.$$

We proceed by rewriting the Bellman equations using the post-decision state variable convention. Let $J_t^x(S_t^x)$ be the value of being in post-decision state S_t^x. Then we can define the relationship between $J_t(S_t)$ and $J_t^x(S_t^x)$ with the following equations

$$J_{t-1}^x(S_{t-1}^x) = \mathbb{E}\{J_t(S_t)|S_{t-1}^x\}, \tag{3}$$

$$J_t(S_t) = \min_{x_t \in \mathcal{X}_{S_t}} (C(S_t, x_t) + \gamma J_t^x(S_t^x)), \tag{4}$$

$$J_t^x(S_t^x) = \mathbb{E}\{J_{t+1}(S_{t+1})|S_t^x\}$$

By substituting Equation (4) into Equation (3), we obtain the Bellman equations around the post-decision state variable

$$J_{t-1}^x(S_{t-1}^x) = \mathbb{E}\left\{\min_{x_t \in \mathcal{X}_{S_t}} (C(S_t, x_t) + \gamma J_t^x(S_t^x)) \middle| S_{t-1}^x\right\}.$$

The important distinction between this post-decision state form and the standard form of the Bellman equations from Equation (1) is the swapping of the expectation and minimum operators. The swap provides computational advantages in that it lets us avoid approximating the expectation explicitly within the optimization problem, and it allows us to control the structure of our value function approximations.

Value Function Approximation.

We estimate our value function using regression methods. In linear regression, the problem is one of estimating a vector to fit a model that will predict a variable using a set of observations. For our model, we wish to estimate the parameter θ using observations that are created from a set of basis functions $\phi_f(S)$, $f \in \mathcal{F}$. The set $\mathcal{F}$ of basis functions allows us to reduce the dimensionality of the state variable to a selected number of features $|\mathcal{F}|$. For example, a basis function $f \in \mathcal{F}$ for our problem might be the interceptor inventory at a SAM site. Using the post-decision state, we can write our value function approximation in a form similar to a standard linear regression model

$$\bar{J}_t^x(S_t^x) = \sum_{f \in \mathcal{F}} \theta_f \phi_f(S_t^x). \tag{5}$$

Our Bellman equations are then expressed as follows

$$\bar{J}_{t-1}^x(S_{t-1}^x) = \mathbb{E}\left\{ \min_{x_t \in \mathcal{X}_{S_t}} (C(S_t, x_t) + \gamma \sum_{f \in \mathcal{F}} \theta_f \phi_f(S_t^x)) \middle| S_{t-1}^x \right\}.$$

We refer to the portion of the Bellman equations inside the expectation operator as the inner minimization problem, or IMP.

IMP.

Consider the IMP of our formulation

$$X_t^\pi(S_t|\theta) = \min_{x_t \in \mathcal{X}_{S_t}} (C(S_t, x_t) + \gamma \sum_{f \in \mathcal{F}} \theta_f \phi_f(S_t^x)). \tag{6}$$

If we assume $\theta_f = 0, \ \forall f \in \mathcal{F}$, then our IMP is simply the minimization of the one period cost function

$$\min_{x_t \in \mathcal{X}_{S_t}} (C(S_t, x_t)),$$

where $C(S_t, x_t) = \mathbb{E}\left\{ \sum_{i \in \mathcal{A}} v_i(A_t - \hat{A}_{t+1,i}) \right\}$. Consider a problem instance of one city and one SAM battery where two attacker missiles have been fired at the city. Then $C(S_t, x_t) = \mathbb{E}\{v(A_t - \hat{A}_{t+1})\} = vA_t - v\psi(x_t)$. Since vA_t is a constant, we reduce the IMP to

$$\min_{x_t \in \mathcal{X}_{S_t}} (-v(1 - q^{x_{t1}})(1 - q^{x_{t2}})).$$

Influencing our pending choice of solution methodology and its corresponding efficacy, we note the following theorem.

Theorem 1 *The integer relaxation of the IMP is not a convex optimization problem.*

Proof. By contradiction, assume that the IMP is a convex optimization problem. Then $\min_{x_t \in \mathcal{X}_{S_t}} (-v(1-q^{x_{t1}})(1-q^{x_{t2}}))$ is convex on $\mathcal{X}_{S_t} = \{x_t : x_{t1} + x_{t2} \leq R_t\}$ and so the Hessian $H(x_{t1}, x_{t2})$ of $-v(1-q^{x_{t1}})(1-q^{x_{t2}})$ is positive definite on $\mathcal{X}_{S_t}$.

Consider the Hessian when $R_t = 10$. This instance yields

$$H(x_{t1}, x_{t2}) = \begin{bmatrix} 2.59029 \times 0.2^{x_{t1}}(1-0.2^{x_{t2}}) & -2.59029 \times 0.2^{x_{t1}+x_{t2}} \\ -2.59029 \times 0.2^{x_{t1}+x_{t2}} & 2.59029 \times 0.2^{x_{t2}}(1-0.2^{x_{t1}}) \end{bmatrix}.$$

Now, consider that a feasible solution in $\mathcal{X}_{S_t} : x_{t1} = 0, x_{t2} = 1$, results in

$$H(0,1) = \begin{bmatrix} 2.07223 & -0.518058 \\ -0.518058 & 0 \end{bmatrix}.$$

Since $(2.07223)(0) - (-.518058)^2 < 0$, by Lemma 3.3.11 of Bazaara et al. [49], $H(0,1)$ is not positive definite. $\square$

Due to Theorem 1, both the IMP and its integer relaxation are nonconvex and hence lack an exact solution method other than exhaustive enumeration of $\mathcal{X}_{S_t}$. Therefore, we invoke MATLAB's genetic algorithm solver to provide an additional solution method for the IMP during our analysis.

Algorithmic Strategy.

Approximate policy iteration (API) is an algorithmic strategy that seeks to approximate the value of a fixed policy within an inner loop and then use that value to update the policy. To employ this strategy, we need a method of approximating a policy. Since the value of our policy depends on a value function approximation based on a linear model (see Equation (5)), we can incorporate a temporal difference (TD) learning algorithm into the API framework. TD algorithms represent an important class of ADP solution techniques and have evolved to include a number of variations.

Least squares temporal differences (LSTD) collects batches of temporal differences and then uses least squares regression to find the best fit. Thus, we can use LSTD to evaluate the approximate value of a policy which we then use to improve the policy iteratively. Algorithm 1 shows API-LSTD adapted to our problem.

The algorithm consists of K policy evaluation loops and N policy improvement loops. After initializing a θ vector as the representation of a base policy, the policy evaluation loop begins by generating a random post-decision state. Once the value $\phi(S^x_{t-1,k})$ is recorded, we simulate forward to the next pre-decision state and select the best decision as per Equation (6). We record the associated expect cost $C(S_{t,k}, x_t)$ and basis function evaluations of the post-decision state, $\phi(S^x_{t,k})$. We obtain K temporal difference sample realizations where the kth temporal difference given the parameter vector θ^n is $(C(S_{t,k}, x_t) + \gamma\phi(S^x_{t,k})^T\theta^n) - \phi(S^x_{t,k-1})^T\theta^n$.

The policy improvement loop of the algorithm begins once K temporal difference sample realizations have been collected. We compactly denote basis function matrices and cost vectors as follows. Let

$$\Phi_{t-1} \triangleq \begin{bmatrix} \phi(S^x_{t-1,1})^\top \\ \vdots \\ \phi(S^x_{t-1,K})^\top \end{bmatrix}, \quad \Phi_t \triangleq \begin{bmatrix} \phi(S^x_{t,1})^\top \\ \vdots \\ \phi(S^x_{t,K})^\top \end{bmatrix}, \quad C(S_t) \triangleq \begin{bmatrix} C(S_{t,1}) \\ \vdots \\ C(S_{t,K}) \end{bmatrix},$$

where matrices Φ_{t-1} and Φ_t are rows of basis function evaluations of the sampled post-decision states, and $C(S_t)$ is the cost vector. We perform a least squares regression of Φ_{t-1} and Φ_t against $C(S_t)$ to ensure the sum of the K temporal differences equals zero and calculate $\hat{\theta}$ as per Equation (7). We update our estimate of θ according to Equation (8) where $\alpha_n = \frac{a}{a+n-1}, a \in (0, \infty)$ denotes our smoothing function. The smoothing function controls the rate of convergence of the algorithm. Higher values of the parameter a slow the rate at which α_n drops to zero. Smoothing θ completes one policy improvement step.

Algorithm 1 LSTD algorithm for infinite horizon problems using basis functions [46]

Initialization:

Initialize θ^0.

Set $n = 1$.

Set the initial policy:

$$X_t^\pi(S_t|\theta^{n-1}) = \underset{x_t \in \mathcal{X}_{S_t}}{\operatorname{argmin}}(C(S_t, x_t) + \gamma\phi(S^{M,x}(S_t, x_t))^T\theta^{n-1})$$

Do for $n = 1, ..., N$:

Do for $k = 1, ..., K$:

Generate random post-decision state $S_{t-1,k}^x$.

Record $\phi(S_{t-1,k}^x)$

Sample W_t.

Compute the next pre-decision state $S_{t,k}$.

Compute the action $x_t = X_t^\pi(S_{t,k}|\theta^{n-1})$.

Compute post-decision state $S_{t,k}^x = S^{M,x}(S_{t,k}, x_t)$.

Record $C(S_{t,k}, x_t)$.

Record $\phi(S_{t,k}^x)$.

Update θ^n and the policy:

$$\hat{\theta} = [(\Phi_{t-1} - \gamma\Phi_t)^T(\Phi_{t-1} - \gamma\Phi_t)]^{-1}(\Phi_{t-1} - \gamma\Phi_t)^T C_t \tag{7}$$

$$\theta^n = \alpha_n\hat{\theta} + (1 - \alpha_n)\theta^{n-1} \tag{8}$$

Return $X_t^\pi(S_t|\theta^N)$ and θ^N.

IV. Computational Tests

In this chapter, we propose a baseline missile defense scenario from which we create four unique test scenarios. We solve each scenario exactly utilizing classical dynamic programming methods and approximately by employing the ADP solution methodology described in the previous chapter. Moreover, for each scenario, we conduct computational experiments to identify the best performance settings for the ADP algorithm. Further, we compare the optimal and ADP missile defense policies for selected subsets of the state space to enhance our understanding of the proposed methodology.

4.1 Scenario

We present a BM defense scenario consisting of three cities defended by two SAM sites. The SAMs are located at the first and third cities and are positioned in such a way as to overlap the second, or "middle" city. That is, the first SAM can defend the first and second cities while the second SAM can defend the second and third cities. City values are 1, 10, and 5 units, respectively. Each SAM site has a preallocation of 10 interceptors and a firing limit of four interceptors per salvo. We set $q = 0.1$ and consider an attacker that can fire up to three missiles per salvo across all cities.

From this baseline scenario, we developed four test scenarios by varying two of the problem features. The first feature we varied was the expected duration of the conflict, in terms of the number of expected attack salvos, as indicated by γ. We chose two γ-values, 0.5 and 0.8, to explore the impact the expected number of salvos would have on the policies. The second feature we varied was the battlefield damage assessment (BDA) capabilities of the attacker. BDA settings are either zero, indicating the attacker has no BDA capabilities, or one, indicating the attacker can identify the

Table 1. Scenarios

	Problem Features: Expected Conflict Duration	Problem Features: Attacker BDA
I	Long($\gamma = 0.8$)	Not Performed (BDA=0)
II	Long($\gamma = 0.8$)	Performed (BDA=1)
III	Short($\gamma = 0.5$)	Not Performed (BDA=0)
IV	Short($\gamma = 0.5$)	Performed (BDA=1)

status of each city prior to launching a missile salvo. For both BDA settings, we utilize a multinomial probability distribution to characterize the attack salvo. We assume the attacker fires 1, 2, or 3 missiles in a salvo with equal probability; given this outcome, when no BDA capability is present (i.e., BDA=0), the probability an attacker fires at a city is the proportion of the city's value to the total value of all cities. When BDA capability is present (i.e., BDA=1), the probability an attacker fires at a city is the proportion of the city's value to the total value of the *remaining* cities. As an example, if City 1 and City 2 are alive (and City 3 is dead) then when BDA=0, the multinomial probability distribution is parameterized by the tuple (1/16, 10/16, 5/16). When BDA=1, the multinomial probability distribution is parameterized by the tuple (1/11, 10/11, 0). Table 1 shows the problem feature settings for each test Scenario.

4.2 Experimental Design

For each of the four test scenarios, we wish to determine the best parameter settings for Algorithm 1. We focus on parameters $N, K, \phi(S)$, and a. Table 2 shows the 3-level, 4-factor experimental design while Table 3 shows the set of features for each design level of the $\phi(S)$ factor. The levels for each factor were chosen based on initial experimental runs of the model.

For each test scenario, we ran a full factorial experiment for 30 random number

Table 2. Experimental Design

N	K	$\phi(S)$	a
5	50	1	0.01
10	100	2	0.5
15	150	3	1.0

Table 3. Basis Function Features

$\phi(S)$	$\phi_0(S)$	$\phi_1(S)$	$\phi_2(S)$	$\phi_3(S)$	$\phi_4(S)$	$\phi_5(S)$
1	1	A_t	R_t^x			
2	1	A_t	R_t^x	A_t^x		
3	1	R_t^x	A_t^x	$(R_t^x)^2$	$(A_t^x)^2$	$R_t^x A_t^x$

seeds for a total of 2430 runs per scenario. For each run, we recorded the mean of the optimality gap for the states containing the full complement of cities and interceptors. For each scenario, we chose the settings that yielded the lowest mean value out of all runs. Table 4 shows the lowest mean optimality gap of the experimental runs for each scenario. The best performing algorithmic settings for each scenario are shown in Table 6.

4.3 Analysis

Due the size of the state space, $|\mathcal{S}| = 16094$, we examined subsets of $\mathcal{S}$ to gain insight into the performance of the ADP algorithm. Each of these subsets can be thought of as a vignette that represents a different starting point for the defender. Because the most interesting problem features involve the overlapping SAM coverage of the cities, we only consider vignettes in which all cities are alive. Instead of varying the city status, we consider a small subset of possible interceptor inventories for both SAM sites (located at City1 and City 3). We consider the following five interceptor inventory levels for R_t: (10,0,10), (8,0,2), (5,0,5), (2,0,8) and (2,0,2). Thus, there are five vignettes. Each vignette represents a decision epoch of interest–a collection of system states at which all cities are alive (as indicated by A_t =(1,1,1)), a number of

Table 4. Experimental Results for Scenario II

Run	N	K	$\phi(S)$	a	Scenario I	Scenario II	Scenario III	Scenario IV
1	15	50	0.01	1	14.86%	15.58%	52.52%	52.76%
2	5	150	0.01	1	14.86%	15.56%	52.52%	52.77%
3	10	150	1	1	18.78%	16.66%	52.52%	52.76%
4	15	50	0.5	1	17.27%	15.96%	52.52%	52.76%
5	15	100	0.5	2	19.05%	15.56%	56.31%	56.68%
6	10	100	0.5	2	19.05%	15.56%	56.31%	56.68%
7	5	150	0.5	3	15.29%	31.15%	35.07%	152.76%
8	15	150	1	2	19.04%	15.56%	52.76%	52.97%
9	5	100	0.5	3	18.96%	32.16%	35.70%	142.76%
10	15	150	0.5	1	18.89%	16.78%	52.52%	52.76%
11	15	150	1	1	18.89%	16.66%	52.52%	52.76%
12	15	100	0.01	2	19.00%	15.56%	53.74%	52.76%
13	15	100	1	3	19.97%	52.04%	35.81%	190.44%
14	5	100	0.5	2	19.00%	15.56%	56.31%	53.01%
15	10	150	0.01	3	19.53%	23.61%	38.02%	59.18%
16	5	50	0.5	1	14.86%	15.58%	52.52%	52.76%
17	10	50	0.01	3	15.47%	18.52%	47.19%	43.49%
18	10	150	1	3	18.06%	56.30%	33.69%	175.52%
19	15	150	0.5	3	16.23%	45.20%	33.34%	155.69%
20	10	150	0.01	1	14.86%	15.56%	52.52%	52.76%
21	5	150	0.01	3	19.47%	23.83%	37.96%	59.15%
22	15	150	0.01	2	14.97%	15.56%	52.68%	52.97%
23	10	100	0.01	3	19.34%	21.75%	34.97%	51.97%
24	5	150	0.5	1	18.78%	16.66%	52.52%	52.76%
25	15	50	1	2	19.05%	15.44%	56.31%	56.76%
26	5	50	0.5	2	15.00%	15.63%	52.52%	52.76%
27	5	100	0.01	3	19.35%	21.75%	34.97%	51.63%
28	5	100	0.01	1	14.86%	15.58%	52.52%	52.76%
29	10	100	1	2	18.88%	15.63%	56.31%	52.97%
30	15	50	1	1	18.79%	16.57%	52.53%	52.77%
31	15	100	0.5	1	19.05%	16.66%	52.52%	52.76%
32	15	150	0.5	2	19.04%	15.56%	52.76%	53.14%
33	10	50	1	1	17.27%	15.96%	52.52%	52.76%
34	10	150	0.5	2	19.04%	15.63%	52.71%	56.68%
35	15	50	0.01	2	14.86%	16.03%	52.68%	56.68%
36	15	50	0.5	3	20.02%	33.72%	40.35%	129.90%
37	5	150	1	3	16.94%	42.98%	35.19%	159.57%
38	5	50	1	3	16.05%	36.91%	35.31%	137.41%
39	10	100	1	3	19.79%	51.98%	34.95%	163.32%
40	15	100	0.01	3	19.34%	21.77%	34.99%	51.97%
41	10	50	0.01	1	14.86%	15.58%	52.52%	52.76%

Table 5. Experimental Results (Cont.) for Scenario II

Run	N	K	$\phi(S)$	a	Scenario I	Scenario II	Scenario III	Scenario IV
42	5	50	1	1	14.93%	15.56%	52.52%	52.76%
43	10	150	0.01	2	14.97%	15.56%	52.68%	52.97%
44	10	150	0.5	1	18.89%	16.66%	52.52%	52.76%
45	5	100	1	1	19.06%	16.59%	52.52%	52.76%
46	10	100	0.5	1	19.05%	16.66%	52.52%	52.76%
47	5	50	0.01	3	15.46%	18.51%	47.22%	43.40%
48	10	50	0.5	2	17.69%	16.65%	52.76%	52.97%
49	10	100	0.01	1	14.86%	15.58%	52.52%	52.76%
50	15	50	0.01	3	15.58%	18.55%	47.18%	43.51%
51	5	100	1	2	18.78%	15.44%	56.31%	52.76%
52	10	100	1	1	19.05%	15.96%	52.52%	52.76%
53	5	150	0.01	2	14.97%	15.56%	52.76%	52.97%
54	15	150	0.01	3	19.54%	23.62%	38.02%	59.18%
55	5	150	1	1	18.78%	16.59%	52.52%	52.76%
56	15	100	0.01	1	14.86%	15.58%	52.52%	52.76%
57	15	150	0.01	1	14.86%	15.56%	52.52%	52.76%
58	5	150	1	2	17.43%	15.44%	52.76%	56.72%
59	10	50	1	3	21.71%	47.74%	36.91%	150.42%
60	10	150	0.5	3	15.99%	40.72%	35.72%	155.37%
61	5	50	0.01	2	14.86%	16.03%	52.68%	56.68%
62	15	100	1	2	19.05%	15.56%	56.31%	56.72%
63	5	50	0.5	3	18.47%	31.17%	39.76%	123.00%
64	15	100	1	1	19.05%	16.66%	52.52%	52.76%
65	10	50	0.5	3	20.51%	33.14%	38.28%	128.25%
66	10	100	0.01	2	19.00%	15.56%	53.74%	52.76%
67	10	150	1	2	18.94%	15.59%	52.76%	52.97%
68	5	150	0.5	2	18.94%	15.44%	52.70%	56.68%
69	10	100	0.5	3	19.65%	45.44%	34.89%	153.50%
70	15	150	1	3	18.80%	59.14%	32.72%	187.76%
71	5	50	1	2	17.69%	15.44%	52.76%	52.76%
72	5	50	0.01	1	14.86%	15.58%	52.52%	52.76%
73	10	50	1	2	19.00%	15.64%	52.76%	53.01%
74	5	100	0.01	2	19.00%	15.56%	53.74%	52.76%
75	15	50	0.5	2	19.05%	15.66%	52.76%	52.97%
76	10	50	0.5	1	14.86%	15.96%	52.52%	52.76%
77	15	100	0.5	3	21.33%	48.32%	36.76%	157.10%
78	5	100	0.5	1	18.89%	16.57%	52.52%	52.76%
79	15	50	1	3	22.03%	50.62%	35.36%	152.38%
80	5	100	1	3	19.69%	44.99%	34.64%	161.73%
81	10	50	0.01	2	14.86%	16.03%	52.68%	56.68%

Table 6. Best Algorithm Settings

	N	K	$\phi(S)$	a
I	10	50	1	0.01
II	15	50	2	1.0
III	15	150	3	1.0
IV	5	50	3	0.01

interceptors remain in inventory (as indicated by $R_t = (10, 0, 10)$, for example), and all possible attack vectors (as indicated by $\hat{M}_t$) are represented. We closely examine the differences between the optimal and ADP policies for test scenario II as it has the most representative problem settings and compare its policies across the remaining scenarios.

Test Scenario II Vignettes.

Vignette 1-Full Interceptor Inventories-($R_t = (10, 0, 10)$).

Table 7 shows policy results for both the exact and ADP algorithms for each possible attack vector when the defender has ten interceptors available at each SAM. Overall, when the system is in a state with $A_t = (1, 1, 1)$ and $R_t = (10, 0, 10)$, implementation of the optimal policy results in the expected loss of 4.32. Implementation of the ADP policy results in the expected loss 4.99, for an optimality gap of 15.44%. The overall absolute gap of 0.67 from a total city value of 16 at risk is reasonable.

The ADP policy is noticeably more conservative in assigning multiple interceptors to missiles particularly from the second SAM which reflects a fundamental difference between the policies. The optimal policy for this vignette is to fire one interceptor per missile fired at City 1 and fire two interceptors per missile fired at City 2 or City 3. The optimal policy also assigns interceptors primarily from the first SAM to counter missiles fired at City 2 unless more than two missiles are inbound. The ADP policy is to fire one interceptor per missile fired at City 1 and City 3 and fire two interceptors per missile targeting City 2. Thus, the approximate algorithm appears to undervalue City 3 as compared to the exact algorithm.

We note that the ADP policy agrees with the optimal policy for five out of 19 states, $\hat{M}_t = (0,1,0),(0,2,0),(0,3,0),(1,0,0),(1,1,0)$, resulting in optimality gaps of 13.68%, 12.64%, 10.44%, 13.37%, and 11.86%, respectively. The ADP policy chooses very poorly for three out of 19 states, $\hat{M}_t = (1,0,1),(1,0,2),(2,0,1)$, resulting in optimality gaps of 97.41%, 77.16%, and 95.59%, respectively. These three states correspond to attack vectors that fire at least one missile at City 1 and City 3 but no missiles at City 2. The combined likelihood of these attack vectors is only 0.02, whereas the combined likelihood for the attack vectors associated with the states that are in agreement is 0.47.

One non-intuitive decision resulting from the exact policy occurs when the attack vector is $\hat{M}_t = (0,0,3)$. The optimal policy chooses to counter the first two missiles with one interceptor each while firing two interceptors at the third missile. Based on decisions for other states, we would expect a decision at this state of defending City 3 with two interceptors per missile.

Table 7. Policy Comparison for Test Scenario II, $R_t = (10, 0, 10)$**,** $A_t = (1, 1, 1)$

$\mathbb{P}(\hat{M}_t)$	$\hat{M}_t$			Optimal Policy $X_t^{\pi^*}$						ADP Policy X_t^{π}								
Attack Probability	Attack Vector			SAM 1 Response			SAM 2 Response			SAM 1 Response			SAM 2 Response			J^*	$\bar{J}^*$	Optimality Gap
0.1042	0	0	1	0	0	0	2	0	0	0	0	0	1	0	0	3.86	4.53	17.41%
0.0326	0	0	2	0	0	0	2	2	0	0	0	0	1	1	0	4.32	5.04	16.69%
0.0102	0	0	3	0	0	0	1	1	2	0	0	0	1	1	1	5.01	5.51	9.96%
0.2083	0	1	0	**2**	**0**	**0**	**0**	**0**	**0**	**2**	**0**	**0**	**0**	**0**	**0**	3.88	4.41	13.68%
0.1302	0	1	1	2	0	0	0	2	0	2	0	0	0	1	0	4.32	4.96	14.98%
0.0610	0	1	2	2	0	0	0	2	2	2	0	0	0	1	1	4.81	5.49	14.15%
0.1302	0	2	0	**2**	**2**	**0**	**0**	**0**	**0**	**2**	**2**	**0**	**0**	**0**	**0**	4.34	4.89	12.64%
0.1221	0	2	1	2	2	0	0	0	2	2	2	0	0	0	1	4.82	5.46	13.27%
0.0814	0	3	0	**0**	**2**	**2**	**2**	**0**	**0**	**0**	**2**	**2**	**2**	**0**	**0**	4.86	5.36	10.44%
0.0208	1	0	0	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	3.72	4.22	13.37%
0.0130	1	0	1	1	0	0	2	0	0	1	0	0	0	0	0	4.13	8.16	97.41%
0.0061	1	0	2	1	0	0	2	2	0	1	0	0	0	0	0	4.61	8.16	77.16%
0.0260	1	1	0	**1**	**2**	**0**	**0**	**0**	**0**	**1**	**2**	**0**	**0**	**0**	**0**	4.16	4.65	11.86%
0.0244	1	1	1	1	2	0	0	2	0	1	2	0	0	1	0	4.62	5.22	12.98%
0.0244	1	2	0	1	1	2	1	0	0	0	2	2	0	0	0	4.65	5.54	19.17%
0.0013	2	0	0	1	1	0	0	0	0	0	0	0	0	0	0	3.98	4.72	18.55%
0.0012	2	0	1	1	1	0	2	0	0	0	0	0	0	0	0	4.41	8.63	95.59%
0.0024	2	1	0	1	1	2	0	0	0	0	0	2	0	0	0	4.44	5.11	14.97%
0.0001	3	0	0	1	1	1	0	0	0	0	0	0	0	0	0	4.24	4.72	11.20%
																$\mathbb{E}[J^*]$	$\mathbb{E}[\bar{J}^*]$	$\mathbb{E}$[Gap]
																4.32	4.99	15.44%

Vignette 2-Half Full Interceptor Inventories-($R_t = (5, 0, 5)$).

Table 8 shows policy results for both the exact and ADP algorithms for each possible attack vector when the defender has five interceptors available at each SAM site. Overall, when the system is in a state with $A_t = (1, 1, 1)$ and $R_t = (5, 0, 5)$, implementation of the optimal policy results in the expected loss of 7.13. Implementation of the ADP policy results in the expected loss 7.78, for an optimality gap of 9.11%. The overall absolute gap of 0.65 from a total city value of 16 at risk is slightly less than the first vignette.

With the reduced inventories, the optimal policy switches to firing one interceptor per missile fired at City 1 *and* City 3 and firing two interceptors per missile fired at City 2 while still preferring to assign interceptors from the first SAM. This policy is the same one closely followed by the ADP policy for the first vignette, i.e., $R_t = (10, 0, 10)$. In fact, the ADP policy for the second vignette, i.e., $R_t = (5, 0, 5)$ is identical to the ADP policy from the first. This change in the optimal policy accounts for a higher number of the identical decisions between the two policies. In this vignette, the ADP policy agrees with the optimal policy for 13 out of 19 states, but still performs the worst for the same three attack vectors as in the previous vignette, (1,0,1),(1,0,2),(2,0,1).

The optimal policy exhibits counterintuitive behavior for the state containing attack vector $\hat{M}_t = (0, 3, 0)$. Instead of firing two interceptors at each missile, it fires two at the first and second missiles, but only one at the third. Conversely, the ADP policy fires two interceptors at each missile, a more intuitive decision. Also observed in this vignette, both the optimal and ADP policies do not defend City 1 against all attack vectors. Out of the 10 attack vectors that target City 1, the optimal policy fires protective interceptors against all except two–$\hat{M}_t = (2, 1, 0), (3, 0, 0)$–while the ADP policy fires interceptors against only five.

Table 8. Policy Comparison for Test Scenario II, $R_t = (5, 0, 5)$**,** $A_t = (1, 1, 1)$

$\mathbb{P}(\hat{M}_t)$	$\hat{M}_t$			Optimal Policy $X_t^{\pi^*}$						ADP Policy X_t^{π}								
Attack Probability	Attack Vector			SAM 1 Response			SAM 2 Response			SAM 1 Response			SAM 2 Response			J^*	$\bar{J}^*$	Optimality Gap
0.1042	0	0	1	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	6.38	6.88	7.97%
0.0326	0	0	2	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	7.04	7.53	6.96%
0.0102	0	0	3	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**1**	**1**	**1**	7.71	8.16	5.75%
0.2083	0	1	0	**2**	**0**	**0**	**0**	**0**	**0**	**2**	**0**	**0**	**0**	**0**	**0**	6.41	6.98	8.88%
0.1302	0	1	1	**2**	**0**	**0**	**0**	**1**	**0**	**2**	**0**	**0**	**0**	**1**	**0**	7.09	7.66	8.06%
0.0610	0	1	2	**2**	**0**	**0**	**0**	**1**	**1**	**2**	**0**	**0**	**0**	**1**	**1**	7.77	8.36	7.57%
0.1302	0	2	0	**2**	**2**	**0**	**0**	**0**	**0**	**2**	**2**	**0**	**0**	**0**	**0**	7.21	7.86	9.04%
0.1221	0	2	1	**2**	**2**	**0**	**0**	**0**	**1**	**2**	**2**	**0**	**0**	**0**	**1**	7.96	8.62	8.32%
0.0814	0	3	0	0	2	2	1	0	0	0	2	2	2	0	0	8.18	8.87	8.45%
0.0208	1	0	0	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	6.10	6.62	8.57%
0.0130	1	0	1	1	0	0	1	0	0	1	0	0	0	0	0	6.78	9.92	46.44%
0.0061	1	0	2	1	0	0	1	1	0	1	0	0	0	0	0	7.44	9.92	33.36%
0.0260	1	1	0	**1**	**2**	**0**	**0**	**0**	**0**	**1**	**2**	**0**	**0**	**0**	**0**	6.84	7.48	9.39%
0.0244	1	1	1	**1**	**2**	**0**	**0**	**1**	**0**	**1**	**2**	**0**	**0**	**1**	**0**	7.54	8.23	9.05%
0.0244	1	2	0	1	1	2	1	0	0	0	2	2	0	0	0	7.74	8.53	10.30%
0.0013	2	0	0	1	1	0	0	0	0	0	0	0	0	0	0	6.50	6.95	6.87%
0.0012	2	0	1	1	1	0	1	0	0	0	0	0	0	0	0	7.19	10.25	42.69%
0.0024	2	1	0	**0**	**0**	**2**	**0**	**0**	**0**	**0**	**0**	**2**	**0**	**0**	**0**	7.23	7.68	6.20%
0.0001	3	0	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	6.56	6.95	5.96%
																$\mathbb{E}[J^*]$	$\mathbb{E}[\bar{J}^*]$	$\mathbb{E}$[Gap]
																7.13	7.78	9.11%

Vignette 3-SAM 1 Inventory High, SAM 2 Inventory Low-($R_t = (8, 0, 2)$).

Table 9 shows policy results for both the exact and ADP algorithms for each possible attack vector when the defender has eight interceptors available at the first SAM but only two available at the second. Overall, when the system is in a state with $A_t = (1, 1, 1)$ and $R_t = (8, 0, 2)$, implementation of the optimal policy results in the expected loss of 7.49. Implementation of the ADP policy results in the expected loss 7.99, for an optimality gap of 6.63%. The overall absolute gap of 0.50 from a total city value of 16 at risk is the lowest among all vignettes.

With a few exceptions, the optimal policy is the same as it was for the second vignette, one interceptor to one missile for City 1 and City 3, two interceptors to one missile for City 2. The ADP policy is identical to that of the first two vignettes with an exception for the state with attack vector $\hat{M}_t = (0, 0, 3)$. In this case, City 3 is being attacked by more missiles than SAM 2 has interceptors, and the ADP policy correctly chooses to save its interceptors for a possible future engagement.

The optimal policy again exhibits counterintuitive behavior for the state containing attack vector $\hat{M}_t = (0, 3, 0)$ choosing for this vignette to fire one interceptor each at the first two missiles but fire two interceptors at the second missile. As before, the ADP policy still fires two interceptors per missile. The optimal policy also acts counterintuitively for the attack vector $\hat{M}_t = (1, 2, 0)$. For this state, the policy fires one interceptor at the first missile, one interceptor at the second missile, and two interceptors at the third. In the second vignette, with an inventory of five interceptors at SAM 2, the exact policy fires an additional interceptor at the second missile from SAM 2. With only two interceptors available at SAM 2 in this vignette, the optimal policy does not add the second interceptor.

Also, for the attack vector $\hat{M}_t = (2, 1, 0)$, owing to an inventory of eight interceptors instead of five at SAM 1, the optimal policy chooses to defend City 1 instead of letting it be destroyed. However, the eight interceptors are not enough for the optimal policy to choose to defend City 1 when the attack vector is $\hat{M}_t = (3, 0, 0)$. This change in the optimal policy from the last vignette results in a policy-to-policy match of 12 out of 19 states with the ADP still making the worst decisions for the same three attack vectors previously identified.

Table 9. Policy Comparison for Test Scenario II, $R_t = (8, 0, 2)$, $A_t = (1, 1, 1)$

$\mathbb{P}(\hat{M}_t)$ Attack Probability	$\hat{M}_t$ Attack Vector	Optimal Policy $X_t^{\pi^*}$ SAM 1 Response	Optimal Policy $X_t^{\pi^*}$ SAM 2 Response	ADP Policy X_t^{π} SAM 1 Response	ADP Policy X_t^{π} SAM 2 Response	J^*	$\bar{J}^*$	Optimality Gap
0.1042	0 0 1	**0 0 0**	**1 0 0**	**0 0 0**	**1 0 0**	6.92	7.23	4.56%
0.0326	0 0 2	**0 0 0**	**1 1 0**	**0 0 0**	**1 1 0**	8.01	8.25	3.02%
0.0102	0 0 3	**0 0 0**	**0 0 0**	**0 0 0**	**0 0 0**	9.23	9.54	3.35%
0.2083	0 1 0	**2 0 0**	**0 0 0**	**2 0 0**	**0 0 0**	6.63	7.07	6.73%
0.1302	0 1 1	**2 0 0**	**0 1 0**	**2 0 0**	**0 1 0**	7.53	7.87	4.59%
0.0610	0 1 2	**2 0 0**	**0 1 1**	**2 0 0**	**0 1 1**	8.60	8.87	3.15%
0.1302	0 2 0	**2 2 0**	**0 0 0**	**2 2 0**	**0 0 0**	7.35	7.89	7.45%
0.1221	0 2 1	**2 2 0**	**0 0 1**	**2 2 0**	**0 0 1**	8.24	8.70	5.59%
0.0814	0 3 0	1 1 2	0 0 0	0 2 2	2 0 0	8.41	9.36	11.34%
0.0208	1 0 0	**1 0 0**	**0 0 0**	**1 0 0**	**0 0 0**	6.34	6.76	6.53%
0.0130	1 0 1	1 0 0	1 0 0	1 0 0	0 0 0	7.26	9.81	35.18%
0.0061	1 0 2	1 0 0	1 1 0	1 0 0	0 0 0	8.35	9.81	17.56%
0.0260	1 1 0	**1 2 0**	**0 0 0**	**1 2 0**	**0 0 0**	7.02	7.50	6.78%
0.0244	1 1 1	**1 2 0**	**0 1 0**	**1 2 0**	**0 1 0**	7.91	8.31	5.01%
0.0244	1 2 0	1 1 2	0 0 0	0 2 2	0 0 0	7.96	8.59	7.97%
0.0013	2 0 0	1 1 0	0 0 0	0 0 0	0 0 0	6.72	7.18	6.96%
0.0012	2 0 1	1 1 0	1 0 0	0 0 0	0 0 0	7.62	10.28	34.90%
0.0024	2 1 0	1 1 2	0 0 0	0 0 2	0 0 0	7.44	7.82	5.11%
0.0001	3 0 0	**0 0 0**	**0 0 0**	**0 0 0**	**0 0 0**	6.86	7.18	4.72%
						$\mathbb{E}[J^*]$	$\mathbb{E}[\bar{J}^*]$	$\mathbb{E}[\text{Gap}]$
						7.49	7.99	6.63%

Vignette 4-SAM 1 Inventory Low, SAM 2 Inventory High-($R_t = (2, 0, 8)$).

Table 10 shows policy results for both the exact and ADP algorithms for each possible attack vector when the defender has two interceptors available at the first SAM site and eight available at the second. Overall, when the system is in a state with $A_t = (1, 1, 1)$ and $R_t = (2, 0, 8)$, implementation of the optimal policy results in the expected loss of 7.12. Implementation of the ADP policy results in the expected loss 7.89, for an optimality gap of 10.89%. The overall absolute gap of 0.77 from a total city value of 16 at risk is the highest gap so far but still of good quality.

The optimal policy is still of the same form as the last two vignettes; however, it now assigns more interceptors from the second SAM in defense of City 2. In other words, the number of interceptors being fired at each missile is roughly the same; however, the split of interceptors fired from the SAMs is flipped. The optimal policy for this vignette remains counterintuitive for the attack vector $\hat{M}_t = (0, 3, 0)$ in a similar manner as the first three vignettes. Additionally, the optimal policy no longer defends City 1 for attack vectors $\hat{M}_t = (2, 0, 1), (2, 1, 0)$, and $(3, 0, 0)$, allowing City 1 to be destroyed. An example of this policy is observed for attack vector $\hat{M}_t = (2, 1, 0)$. The optimal policy chooses to have the first SAM fire both of its remaining interceptors at the missile inbound to City 2 leaving City 1 undefended and letting the SAM 2 conserve its inventory to protect the more valuable City 2 from future attack salvos.

The ADP policy observed in this vignette differs for the first time from the previous vignettes. Although it has the same general interceptor-to-missile policy, instead of utilizing SAM 2 to provide additional defense for City 2, it conserves that SAM's inventory at the expense of depleting SAM 1. For example, for the state with attack vector $\hat{M}_t = (0, 1, 0)$, instead of firing one interceptor from each SAM as the optimal policy does, the approximate policy fires two interceptors from SAM 1 and none from SAM 2, leaving City 1 defenseless against future attacks. Similarly, for attack vector $\hat{M}_t = (1, 1, 0)$, the ADP policy fires two interceptors from SAM 1 at the missile inbound to City 2 while firing none from SAM 2, thus choosing to conserve SAM 2 interceptors for future use over protecting City 1 during from the current attack.

As seen previously, the ADP policy chooses poorly for the three identified states, but for this vignette it also performs poorly for an additional attack vector of $\hat{M}_t = (1, 2, 0)$. The ADP policy fires two interceptors from both SAM sites at only the first inbound missile to City 2 ensuring that both City 1 and City 2 are destroyed. The optimal and ADP policies match exactly for six out of 19 states.

Table 10. Policy Comparison for Test Scenario II, $R_t = (2, 0, 8)$**,** $A_t = (1, 1, 1)$

$\mathbb{P}(\hat{M}_t)$	$\hat{M}_t$			Optimal Policy $X_t^{\pi^*}$						ADP Policy X_t^{π}								
Attack Probability	Attack Vector			SAM 1 Response			SAM 2 Response			SAM 1 Response			SAM 2 Response			J^*	$\bar{J}^*$	Optimality Gap
0.1042	0	0	1	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	6.36	6.96	9.45%
0.0326	0	0	2	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	6.98	7.59	8.78%
0.0102	0	0	3	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**1**	**1**	**1**	7.58	8.18	7.93%
0.2083	0	1	0	1	0	0	1	0	0	2	0	0	0	0	0	6.40	7.01	9.47%
0.1302	0	1	1	1	0	0	1	1	0	2	0	0	0	1	0	7.08	7.65	7.95%
0.0610	0	1	2	1	0	0	1	1	1	2	0	0	0	1	1	7.75	8.32	7.40%
0.1302	0	2	0	0	1	0	2	1	0	0	2	0	2	0	0	7.21	7.80	8.29%
0.1221	0	2	1	0	1	0	2	1	1	0	2	0	2	0	1	7.96	8.49	6.68%
0.0814	0	3	0	0	0	1	1	2	1	0	0	2	2	2	0	8.18	8.86	8.34%
0.0208	1	0	0	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	6.10	6.68	9.67%
0.0130	1	0	1	1	0	0	1	0	0	1	0	0	0	0	0	6.76	10.03	48.30%
0.0061	1	0	2	1	0	0	1	1	0	1	0	0	0	0	0	7.40	10.03	35.53%
0.0260	1	1	0	1	0	0	2	0	0	0	2	0	0	0	0	6.84	7.67	12.20%
0.0244	1	1	1	1	0	0	2	1	0	0	2	0	0	1	0	7.54	8.30	10.01%
0.0244	1	2	0	1	0	0	2	2	0	0	2	0	2	0	0	7.74	13.36	72.66%
0.0013	2	0	0	1	1	0	0	0	0	0	0	0	0	0	0	6.52	6.97	6.93%
0.0012	2	0	1	0	0	0	1	0	0	0	0	0	0	0	0	7.18	10.31	43.48%
0.0024	2	1	0	**0**	**0**	**2**	**0**	**0**	**0**	**0**	**0**	**2**	**0**	**0**	**0**	7.22	7.67	6.29%
0.0001	3	0	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	6.54	6.97	6.62%
																$\mathbb{E}[J^*]$	$\mathbb{E}[\bar{J}^*]$	$\mathbb{E}$[Gap]
																7.12	7.89	10.89%

Vignette5-Low Interceptor Inventories-($R_t = (2, 0, 2)$).

Table 11 shows policy results for both the exact and ADP algorithms for each possible attack vector when the defender has only two interceptors available at each SAM. Overall, when the system is in a state with $A_t = (1, 1, 1)$ and $R_t = (2, 0, 2)$, implementation of the optimal policy results in the expected loss of 10.16. Implementation of the ADP policy results in the expected loss 10.94, for an optimality gap of 7.62%. The overall absolute gap of 0.78 from a total city value of 16 at risk is the largest gap among the vignettes.

In this vignette, the optimal policy switches to a one-to-one interceptor-to-missile policy for all cities. Moreover, the optimal policy switches back to having the first SAM provide most of the defense for City 2. The optimal policy also makes decisions in this vignette similar to the decisions of the ADP policy of Vignette 4, wherein City 1 is targeted along with City 2 and City 3. The optimal policy only defends City 1 if it is the only city attacked and even then only if it is attacked with one missile. In all other cases, City 1 is left undefended in order to provide defensive cover for City 2 either immediately or for subsequent attacks. The ADP policy remains the same as observed in earlier vignettes as much as inventories allow. That is, the ADP policy leaves City 1 unprotected so that it can fire two interceptors per incoming missile to City 2.

The optimal and ADP policies match for 6 out of 19 states, but the ADP policy still decides badly at three out of four of the previously mentioned attack vectors $\hat{M}_t = (1, 0, 1), (1, 0, 2), (2, 0, 1)$. The difference in state value is not as extreme as observed in other vignettes since with so few interceptors remaining at each SAM, the cities that are initially protected by the optimal policy will likely be destroyed in one or two more attack salvos.

Table 11. Policy Comparison for Test Scenario II, $R_t = (2, 0, 2)$, $A_t = (1, 1, 1)$

$\mathbb{P}(\hat{M}_t)$	$\hat{M}_t$			Optimal Policy $X_t^{\pi^*}$						ADP Policy X_t^{π}								
Attack Probability	Attack Vector			SAM 1 Response			SAM 2 Response			SAM 1 Response			SAM 2 Response			J^*	$\bar{J}^*$	Optimality Gap
0.1042	0	0	1	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	9.07	9.40	3.59%
0.0326	0	0	2	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	10.20	10.42	2.12%
0.0102	0	0	3	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	10.86	11.31	4.11%
0.2083	0	1	0	1	0	0	0	0	0	2	0	0	0	0	0	9.21	9.83	6.71%
0.1302	0	1	1	1	0	0	0	1	0	2	0	0	0	1	0	10.19	10.64	4.47%
0.0610	0	1	2	1	0	0	0	1	1	2	0	0	0	1	1	11.32	11.94	5.43%
0.1302	0	2	0	1	1	0	0	0	0	0	2	0	2	0	0	10.27	11.65	13.47%
0.1221	0	2	1	1	1	0	0	0	1	0	2	0	1	0	1	11.22	12.04	7.28%
0.0814	0	3	0	0	1	1	1	0	0	0	0	2	1	1	0	11.29	12.10	7.25%
0.0208	1	0	0	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	8.82	9.22	4.52%
0.0130	1	0	1	0	0	0	1	0	0	1	0	0	0	0	0	9.75	11.62	19.11%
0.0061	1	0	2	0	0	0	1	1	0	1	0	0	0	0	0	10.83	11.62	7.25%
0.0260	1	1	0	0	1	0	0	0	0	0	2	0	0	0	0	9.89	10.42	5.44%
0.0244	1	1	1	0	1	0	0	1	0	0	2	0	0	1	0	10.82	11.21	3.64%
0.0244	1	2	0	0	1	1	0	0	0	0	2	0	2	0	0	10.89	14.60	34.12%
0.0013	2	0	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	8.85	9.40	6.28%
0.0012	2	0	1	0	0	0	1	0	0	0	0	0	0	0	0	9.75	11.87	21.76%
0.0024	2	1	0	0	0	1	0	0	0	0	0	2	0	0	0	9.89	10.42	5.44%
0.0001	3	0	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	8.85	9.40	6.28%
																$\mathbb{E}[J^*]$	$\mathbb{E}[\bar{J}^*]$	$\mathbb{E}$[Gap]
																10.16	10.94	7.62%

Analysis Across Scenarios–Vignettes.

Scenario II vs Scenario I.

In Scenario I the expected conflict duration is still long, but BDA is not performed. For three of the five vignettes, $R_t = (10, 0, 10), (8, 0, 2), (2, 0, 2)$, the optimal policy for Scenario I is either identical or nearly identical as for Scenario II. In fact, only one state in the $R_t = (8, 0, 2)$ and $R_t = (2, 0, 2)$ vignettes is different. In both cases, the optimal policy for Scenario I chooses to not defend City 1 whereas it does defend City 1 in Scenario II. For the remaining two vignettes, the optimal policy for Scenario I is slightly more conservative in applying additional interceptors to missiles, as well as being less protective of City 1 as compared to Scenario II.

As compared to Scenario II, in Scenario I the ADP policy fires more interceptors for each vignette with most of those interceptors being used to defend City 1. In Scenario II, the ADP policy did not defend City 1 for a total of 29 states; however, in Scenario I, the ADP policy defends City 1 for 22 of those 29 states. The ADP policy shows the same invariance across vignettes that it showed for Scenario II as well as the same poor performance for the previously noted attack vectors. Overall, the optimal and ADP policies match for the same number of states, albeit for different states in the vignettes.

Scenario II vs Scenario IV.

In Scenario IV the expected conflict duration is short while BDA is still performed. As we would expect given the shorter expected horizon, both the optimal and ADP policies more freely fire interceptors in Scenario IV compared to Scenario II. At times, the optimal policy fires up to three interceptors per missile when defending City 2. Table 12 shows the total number of additional interceptors fired in each vignette of Scenario IV for each policy compared to Scenario II. As in Scenario

II, the optimal policy adjusts for the interceptor inventories while the ADP does not. For example, the optimal policy fires more total missiles across all states for vignette $R_t = (10, 0, 10)$, but the roles switch for the remaining vignettes with the ADP policy outfiring the optimal by 25 interceptors for vignette $R_t = (2, 0, 2)$.

Table 12. Number of additional interceptors fired

	Policy	
Vignette	Optimal	ADP
$R_t = (10, 0, 10)$	24	37
$R_t = (5, 0, 5)$	15	37
$R_t = (8, 0, 2)$	5	30
$R_t = (2, 0, 8)$	18	27
$R_t = (2, 0, 2)$	2	17

Again, the ADP policy performs poorly for the usual three states; however, since the expected horizon is much shorter Scenario IV the ADP's poor performance results in a much larger optimality gap between the exact and approximate values for those states. For example, when the attack vector is $\hat{M}_t = (1, 0, 2)$, the ADP policy fires four interceptors at the first inbound missile to City 3 and none at the rest of the missiles, leaving City 1 and City 3 to be destroyed. This action results in an immediate cost of six units. In contrast, the optimal policy covers down on all three inbound missiles with two interceptors each. This decision means that the optimal policy receives a small expected cost for this salvo.

For Scenario II, the decisions chosen by each policy result in a similar disparity between the one-period costs. Thus, after one salvo, the difference in cost between the two policies is relatively large for both scenarios. However, for Scenario IV there is a much lower probability of cities being destroyed under the optimal policy for the remaining salvos compared to Scenario II, given that there are two expected salvos in Scenario IV compared to five in Scenario II. Hence, the cost difference between the optimal and ADP policies for Scenario IV is unlikely to be reduced. When the horizon is longer, as it is for Scenario II, it is likely that even following the optimal policy will eventually result in the loss of cities. Even though these losses come later in the conflict, losing them still results in closing the gap between the two policies for Scenario II as compared to Scenario IV.

Scenario II vs Scenario III.

In Scenario III, the expected conflict is short and BDA is not performed. Comparing Scenario II to Scenario III, we see great similarity to the comparison between Scenarios II and IV. This similarity is due to the optimal policies for Scenarios III and IV being virtually identical. In fact, both optimal policies fire the same number of interceptors in each vignette, and the decisions themselves are identical with the exception of a lone state in vignette $R_t = (2, 0, 8)$. The ADP policies between Scenarios III and IV differ, but the results are similar.

Across all the scenarios, we observe that the ADP policies are relatively invariant. That is, the decisions made by the ADP policy do not seem to be influenced by the different interceptor inventories in each vignette. This behavior stands in contrast to the behavior of the optimal policies. It is this conflicting behavior that accounts for the variance in the number of states that match across the two policies for a given vignette.

We also note that the setting for γ has a much greater influence on the policies than the setting for BDA. As discussed earlier, the optimal policies of Scenarios I and II vary only slightly in the number of interceptors fired. This change results from the different policy decisions made at 12 states across the five vignettes. Between Scenarios III and IV, the optimal policies are identical with the exception of one state. However, when γ is varied as it is between Scenarios I and III or II and IV, we observe 53 and 44 different decisions, respectively. The ADP policies also exhibits a larger influence of γ, although there are significantly more decisions that differ among the sets of approximate policies.

Analysis Across Scenarios–Full State Space.

Table 13 shows the mean optimality gaps for the optimal and ADP policies in each scenario for the state spaces of each vignette and the entire state space, using the ADP algorithm settings from Table 6

Table 13. Mean Optimality Gaps

	Vignette 1	Vignette 2	Vignette 3	Vignette 4	Vignette 5	All States
I	14.86%	11.67%	8.71%	11.85%	8.89%	10.96%
II	15.44%	9.11%	6.63%	10.89%	7.62%	7.74%
III	32.72%	20.13%	20.42%	31.20%	17.88%	22.10%
IV	43.40%	29.48%	33.00%	36.47%	21.52%	15.51%

Over all scenarios, the mean optimality gap is lower when taken over the entire state space. This reflects the good performance of the ADP algorithm for states with smaller inventories and fewer surviving cities.

Figures 2 through 5 show the number of states in the entire state space that correspond to the absolute value of the difference between the optimal and approximate values of each state for each scenario. We note that the approximate value for a majority of states for each scenario falls within 0.5 units of the optimal value.

	BDA=0	
	$\lvert J^* - \bar{J}^* \rvert$	No. of States
	0	1324
	0.5	8615
$\gamma = 0.8$	1.0	4527
	1.5	346
	2.0	142
	> 2.0	1139

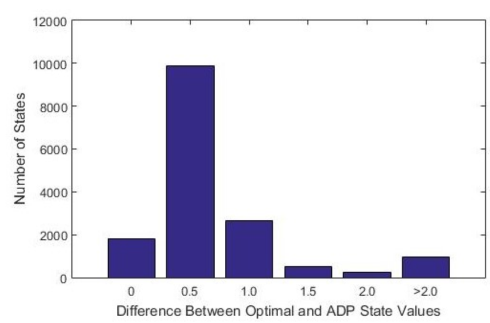


Figure 2. Absolute Value of Difference Scenario I

	BDA=1	
	$\lvert J^* - \bar{J}^* \rvert$	No. of States
	0	1813
	0.5	9899
$\gamma = 0.8$	1.0	2671
	1.5	527
	2.0	239
	> 2.0	944

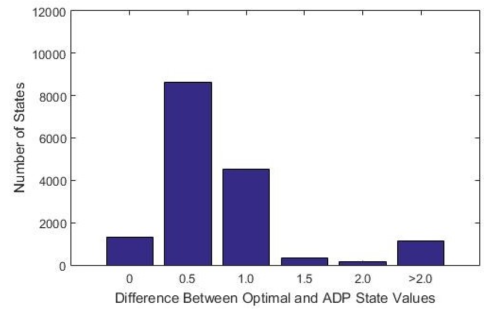


Figure 3. Absolute Value of Difference Scenario II

	BDA=0	
	$\lvert J^* - \bar{J}^* \rvert$	No. of States
	0	938
	0.5	11251
$\gamma = 0.5$	1.0	1908
	1.5	924
	2.0	246
	> 2.0	826

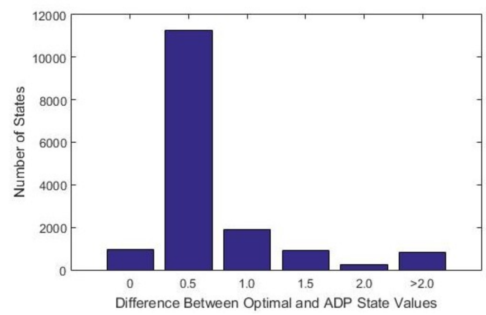


Figure 4. Absolute Value of Difference Scenario III

	BDA=1	
	$\lvert J^* - \bar{J}^* \rvert$	No. of States
	0	1160
	0.5	10277
$\gamma = 0.5$	1.0	2652
	1.5	872
	2.0	383
	> 2.0	749

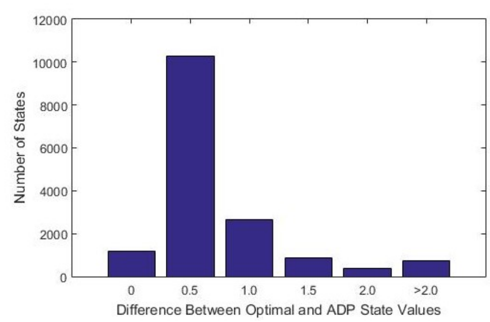


Figure 5. Absolute Value of Difference Scenario IV

V. Conclusions

As the proliferation of offensive and defensive missile systems continues across the world, the optimization of a defensive response to a missile attack remains a valuable endeavor for the U.S. and its allies. Given the likelihood that a BM engagement would involve more than one missile salvo by an attacker, this thesis presented both exact and approximate methods for solving the DWTAP.

Across four test scenarios, when compared to the optimal policy, the ADP policy achieved anywhere from an 8% to 22% expected optimality gap. In addition, for the vast majority of states in all scenarios, the state values for the ADP policy fell within 0.5 units of the state values for the exact policy. Analysis also showed that the γ-parameter influenced the fire control policies of both methods more than the attacker's BDA capabilities, and that the ADP policy is invariant unlike the optimal policy, as the interceptor inventories change, resulting in decision agreement between the two policies from 4 to 15 out of 19 states in vignettes within the same scenario.

Future research could explore the performance of a reasonable baseline fire control policy compared with the API-LSTD policy developed in this thesis as well as other ADP algorithms from the literature. Additionally, one could expand the problem beyond the computational tractability of exact methods to larger test scenarios with more cities, greater inventories, and bigger attack salvos to test the scalability of the applied approximate algorithms.

Another improvement to the model would be the development of a "smarter" attacker. In addition to a BDA capability, an attacker could be enabled with knowledge of the defender's interceptor inventories. Varying the quality of both these intelligence capabilities over a wider range of settings could provide insight as well. Ultimately, a learning policy for the attacker, one that responds to the defender's policy, would provide the most realistic matchup.

Many of the assumptions in this thesis could also be eliminated through the inclusion of multiple missile and interceptor types, introduction of a SAM reload capability, and/or the addition of the partial destruction of cities. More significant model changes could include incorporating subsequent targeting of missed missiles within the same epoch, i.e., a shoot-look-shoot policy, the development of a more complete IADS structure, and the targeting of IADS nodes by the attacker.

V. Appendix

Determination of Fire Control Policies via Approximate Dynamic Programming

Capt Michael T. Davis, Co-advisors: Lt Col Matthew J. Robbins, Ph.D., LTC Brian J. Lunday, Ph.D.

Background

- Over 30 countries have inventories of theater ballistic missiles while another 50 employ multiple launch rocket systems.
- Ballistic missile attacks typically extend over many salvos of missile launches by the offense.
- Defender must decide how many interceptors to employ against the missiles within the current salvo and how many to conserve for subsequent salvos.
- Given a set of cities with values, a set of SAM batteries with corresponding city-covering capabilities, and a fixed number of interceptor missiles at each battery, identify the best interceptor allocation policy for a defender.

Methodology

- We formulate an infinite horizon, discrete time stochastic Markov decision process model of the dynamic weapon-target assignment problem.
- We determine optimal policies for small problem instances to obtain insight concerning policy structure.
- For larger instances, the high dimensionality of the state space makes solving the MDP computationally intractable.
- We develop an approximate policy iteration algorithm with Bellman error minimization to determine near-optimal policies.
- Within the least-squares temporal differences policy evaluation step, we use a modified version of the Bellman equations that is based on the post-decision state variable.

$$J^x_{t-1}(S^x_{t-1}) = \mathbb{E}\left\{\min_{x_t \in \mathcal{X}_{S_t}} \left(C(S_t, x_t) + \gamma J^x_t(S^x_t)\right) \Big| S^x_{t-1}\right\}$$

- A linear architecture is used to map features of the problem to a set of basis functions allowing us to approximate the value function.

$$\bar{J}^x_t(S^x_t) = \sum_{f \in \mathcal{F}} \theta_f \phi_f(S^x_t)$$

Contact Information

Lt Col Matthew J. Robbins, Ph.D.
LTC Brian J. Lunday, Ph.D.
Department of Operational Sciences, AFIT

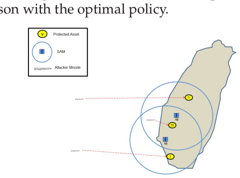
Model

- Objective: Select a fire control policy to minimize the expected total discounted cost of destroyed cities over all epochs
- Time Horizon: $\mathcal{T} = \{1, 2, ..., T\}, T \leq \infty$
- State Space: Consists of the status of each city, the inventory of each SAM site, and the attack "vector". $S_t = (A_t, R_t, \hat{M}_t) \in \mathcal{S}$
- Action Space: Collection of decision vectors, x_t, consisting of the number of interceptors assigned to each missile from each SAM, x_{tij}. $\mathcal{X}_{S_t} = \{x_t : \sum_{j \in \hat{\mathcal{M}}^A_{ti}} x_{tij} \leq R_{ti}, \ \forall\, i \in \mathcal{A}\}$
- Transition Function:
  - City status transitions stochastically: $A_{t+1,i} = \begin{cases} 0 & \text{if } A_{ti} = 0, \\ \hat{A}_{t+1,i}(x_t) & \text{otherwise,} \end{cases} \quad \forall i \in \mathcal{A},$
  - SAM inventory transitions deterministically: $R_{t+1,i} = R_{ti} - \sum_{j \in \hat{\mathcal{M}}^A_{ti}} x_{tij}, \ \forall\, i \in \mathcal{A}$
  - Attack probabilities conditioned on city status: $\mathbb{P}^{\hat{M}_t}(m) = \mathbb{P}(\hat{M}_t = m | A_t)$
- Cost Function: $C(S_t, x_t) = \mathbb{E}\left\{\sum_{i \in \mathcal{A}} v_i(A_t - \hat{A}_{t+1,i}) | S_t, x_t\right\}$, where v_i is value of city $i \in \mathcal{A}$
- Objective: Select a fire control policy to minimize the expected total discounted cost of destroyed cities over all epochs

$$\min_{\pi \in \Pi} \mathbb{E}^\pi \left\{\sum_{t=0}^{T} \gamma^t C(S_t, X^\pi_t(S_t))\right\}$$

Computational Example

Computational results are obtained for four test scenarios which were created from a baseline instance. A designed experiment was run for each test scenario to identify the best performing algorithmic parameter settings. Utilizing these settings, we determine the ADP policy for each test scenario and examine subsets of the state space for comparison with the optimal policy.



Notional baseline instance. Four test scenarios were generated by choosing two γ-values and two settings of battlefield damage assessment (BDA) capabilities for the attacker.

$\mathbb{P}(\hat{M}_t)$ Attack Probability	$\hat{M}_t$ Attack Vector			Optimal Policy $X^{\pi^*}_t$ SAM 1 Response			Optimal Policy $X^{\pi^*}_t$ SAM 2 Response			ADP Policy X^{π}_t SAM 1 Response			ADP Policy X^{π}_t SAM 2 Response			J^*	$\bar{J}^*$	Optimality Gap
0.1042	0	0	1	0	0	0	2	0	0	0	0	0	1	0	0	3.86	4.53	17.41%
0.0326	0	0	2	0	0	0	2	2	0	0	0	0	1	1	0	4.32	5.04	16.69%
0.0102	0	0	3	0	0	0	1	1	2	0	0	0	1	1	1	5.01	5.51	9.96%
0.2083	0	1	0	**2**	**0**	**0**	**0**	**0**	**0**	**2**	**0**	**0**	**0**	**0**	**0**	3.88	4.41	13.68%
0.1302	0	1	1	2	0	0	0	2	0	2	0	0	0	1	0	4.32	4.96	14.98%
0.0610	0	1	2	2	0	0	0	2	2	2	0	0	0	1	1	4.81	5.49	14.15%
0.1302	0	2	0	**2**	**2**	**0**	**0**	**0**	**0**	**2**	**2**	**0**	**0**	**0**	**0**	4.34	4.89	12.64%
0.1221	0	2	1	2	2	0	0	0	2	2	2	0	0	0	1	4.82	5.46	13.27%
0.0814	0	3	0	**0**	**2**	**2**	**2**	**0**	**0**	**0**	**2**	**2**	**2**	**0**	**0**	4.86	5.36	10.44%
0.0208	1	0	0	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	3.72	4.22	13.37%
0.0130	1	0	1	1	0	0	2	0	0	1	0	0	0	0	0	4.13	8.16	97.41%
0.0061	1	0	2	1	0	0	2	2	0	1	0	0	0	0	0	4.61	8.16	77.16%
0.0260	1	1	0	**1**	**2**	**0**	**0**	**0**	**0**	**1**	**2**	**0**	**0**	**0**	**0**	4.16	4.65	11.86%
0.0244	1	1	1	1	2	0	0	2	0	1	2	0	0	1	0	4.62	5.22	12.98%
0.0244	1	2	0	1	1	2	1	0	0	0	2	2	0	0	0	4.65	5.54	19.17%
0.0013	2	0	0	1	1	0	0	0	0	0	0	0	0	0	0	3.98	4.72	18.55%
0.0012	2	0	1	1	1	0	2	0	0	0	0	0	0	0	0	4.41	8.63	95.59%
0.0024	2	1	0	1	1	2	0	0	0	0	0	2	0	0	0	4.44	5.11	14.97%
0.0001	3	0	0	1	1	1	0	0	0	0	0	0	0	0	0	4.24	4.72	11.20%
																$\mathbb{E}[J^*]$	$\mathbb{E}[\bar{J}^*]$	$\mathbb{E}$[Gap]
																4.32	4.99	15.44%

The results from Scenario II, where $\gamma = 0.8$, BDA is performed, $A_t = (1,1,1)$ and $R_t = (10,0,10)$. Five decisions are identical between the two policies, while the mean approximate state value is a modest 0.67 higher than the mean exact state value.

Results and Conclusions

- MDP formulation is computationally tractable for only small instances.
- Across four test scenarios,
  - The ADP solution methods achieve an 8% to 22% mean optimality gap.
  - A majority of approximate state values fall within 0.5 units of the exact state values.
  - The γ-parameter influences fire control policies more than an attacker's BDA capabilities.

	BDA=0		BDA=1	
	$\lvert J^* - \bar{J}^* \rvert$	No. of States	$\lvert J^* - \bar{J}^* \rvert$	No. of States
$\gamma = 0.8$	0	1324	0	1813
	0.5	8615	0.5	9899
	1.0	4527	1.0	2671
	1.5	346	1.5	527
	2.0	142	2.0	239
	> 2.0	1139	> 2.0	944
	$\lvert J^* - \bar{J}^* \rvert$	No. of States	$\lvert J^* - \bar{J}^* \rvert$	No. of States
$\gamma = 0.5$	0	938	0	1160
	0.5	11251	0.5	10277
	1.0	1908	1.0	2652
	1.5	924	1.5	872
	2.0	246	2.0	383
	> 2.0	826	> 2.0	749

Limitations

- Basis functions do not adequately capture importance of SAM inventory or interactions between City 1 and City 3.
- Higher dimensionality problem instances were not considered.

Future Study

- Exploration of the performance of a operationally feasible baseline fire control policy would allow for a more reasonable comparison to API-LSTD policy as well as other ADP algorithms.
- Expansion of the problem beyond computational tractability of exact methods would enable testing of the scalability of approximate methods.
- Incorporation of a "smarter" attacker would more realistically represent a ballistic missile engagement.
- Eliminating current model assumptions, e.g., allowing more missile/interceptor types, would more accurately reflect current offensive/defensive capabilities.

Bibliography

1. George C. Marshall and Claremont Institutes, "Missiles of the World." [Online; accessed 30-July-2015; http://missilethreat.com/missiles-of-the-world/].

2. Global Firepower, "MLRS (Multiple Launch Rocket System) Strength by Country," 2015. [Online; accessed 30-July-2015; http://www.globalfirepower.com/armor-mlrs-total.asp].

3. D. Goure, "Directed energy can defeat massed missile salvoes," 2015. [Online; accessed 30-July-2015; http://www.realcleardefense.com/articles/2015/07/11/directed_energy_can_defeat_massed_missile_salvoes_108212.html].

4. The Jerusalem Post, "Defense Ministry: Upgrades for Iron Dome's operational capabilities a success," 2015. [Online; accessed 30-July-2015; http://www.jpost.com/Israel-News/Defense-Ministry-Upgrades-for-Iron-Domes-operational-capabilities-a-success-407827].

5. C. Davenport, "Patriot, the workhorse of the Pentagon's missile defense system, to get an upgrade." [Online; accessed 4-August-2015; https://www.washingtonpost.com/news/checkpoint/wp/2015/02/20/patriot-the-workhouse-of-the-pentagons-missile-defense-system-to-get-an-upgrade/].

6. Raytheon, "Global Patriot Solutions," 2015. [Online; accessed 4-August-2015; http://www.raytheon.com/capabilities/products/patriot/].

7. C. Harress, "Canadian military buys Israeli Iron Dome missile defense technology," 2015. [Online; accessed 30-July-2015; http://www.ibtimes.

com/canadian-military-buys-israeli-iron-dome-missile-defense-technology-2030063].

8. IBC News Bureau, "Long range surface air missile to be test-fired in India," 2015. [Online; accessed 30-July-2015; https://www.ibcworldnews.com/2015/07/25/long-range-surface-air-missile-to-be-test-fired-in-india/].

9. Reuters, "Turkey open to new bid from China in controversial missile defense tender," 2015. [Online; accessed 30-July-2015; http://national.bgnnews.com/turkey-open-to-new-bid-from-china-in-controversial-missile-defense-tender-haberi/8049].

10. C. Mindock, "US launching missile defense system in Europe, Africa that Russia hates," 2015. [Online; accessed 30-July-2015; http://www.ibtimes.com/us-launching-missile-defense-system-europe-africa-russia-hates-2028707].

11. R. Tomkins, "Raytheon given $2.4B FMS contract for Patriot fire units," 2014. [Online; accessed 4-August-2015; http://www.upi.com/Business_News/Security-Industry/2014/12/23/Raytheon-given-24B-FMS-contract-for-Patriot-fire-units/8031419329356/].

12. PR Newswire, "Republic of Korea upgrades its air and missile defense," 2015. [Online; accessed 4-August-2015; http://www.prnewswire.com/news-releases/republic-of-korea-upgrades-its-air-and-missile-defense-300057545.html].

13. W. Brown, "Lockheed Martin wins $1.5 billion contract for Patriot missile work in Camden," 2015. [Online; accessed 4-August-2015;

http://ualrpublicradio.org/post/lockheed-martin-wins-15-billion-contract-patriot-missile-work-camden].

14. C. Dillow, "U.S. greenlights sale of 600 Patriot missiles to Saudi Arabia," 2015. [Online; accessed 4-August-2015; http://fortune.com/2015/08/01/u-s-patriot-missiles-saudi-arabia/].

15. C. Y. Han, B. J. Lunday, and M. J. Robbins, "A game theoretic model for the optimal disposition of integrated air defense missile batteries," *INFORMS Journal on Computing*, In press.

16. M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2005.

17. R. K. Ahuja, A. Kumar, K. C. Jha, and J. B. Orlin, "Exact and heuristic algorithms for the weapon-target assignment problem," *Operations Research*, vol. 55, no. 6, pp. 1136–1146, 2007.

18. A. M. Madni and M. Andrecut, "Efficient heuristic approach to the weapon-target assignment problem," *Journal of Aerospace Computing, Information, and Communication*, vol. 6, no. 6, pp. 405–414, 2009.

19. Z.-J. Lee, S.-F. Su, and C.-Y. Lee, "Efficiently solving general weapon-target assignment problem by genetic algorithms with greedy eugenics," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 33, no. 1, pp. 113–121, 2003.

20. C. K. Cheong, "Survey of investigations into the missile allocation problem," Master's thesis, Naval Postgraduate School, Monterey CA, 1985.

21. S. Matlin, "A review of the literature on the missile-allocation problem," *Operations Research*, vol. 18, no. 2, pp. 334–373, 1970.

22. A. R. Eckler and S. A. Burr, “Mathematical models of target coverage and missile allocation,” tech. rep., Military Operations Research Society, Alexandria, Va., 1972.

23. A. S. Manne, “A target-assignment problem,” *Operations Research*, vol. 6, no. 3, pp. 346–351, 1958.

24. J. C. Bradford, “Determination of the optimal assignment of a weapon system to several threats,” *Vought Aeronautics, AER-EITM-9*, p. 14, 1961.

25. R. H. Day, “Allocating weapons to target complexes by means of nonlinear programming,” *Operations Research*, vol. 14, no. 6, pp. 992–1013, 1966.

26. J. S. Croucher, “Application of the fundamental theorem of games to an example concerning antiballistic missile defense,” *Naval Research Logistics Quarterly*, vol. 22, no. 1, pp. 197–203, 1975.

27. G. G. denBroeder Jr, R. E. Ellison, and L. Emerling, “On optimum target assignments,” *Operations Research*, vol. 7, no. 3, pp. 322–326, 1959.

28. S. P. Lloyd and H. S. Witsenhausen, “Weapons allocation is NP-complete.,” in *1986 Summer Computer Simulation Conference*, pp. 1054–1058, 1986.

29. B. Xin, J. Chen, J. Zhang, L. Dou, and Z. Peng, “Efficient decision makings for dynamic weapon-target assignment by virtual permutation and tabu search heuristics,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 40, no. 6, pp. 649–662, 2010.

30. X. Zeng, Y. Zhu, L. Nan, K. Hu, B. Niu, and X. He, “Solving weapon-target assignment problem using discrete particle swarm optimization,” in *The Sixth World Congress on Intelligent Control and Automation, 2006.*, vol. 1, pp. 3562–3565, IEEE, 2006.

31. E. Wacholder, “A neural network-based optimization algorithm for the static weapon-target assignment problem,” *ORSA Journal on Computing*, vol. 1, no. 4, pp. 232–246, 1989.

32. R. A. Murphey, “Target-based weapon target assignment problems,” in *Nonlinear Assignment Problems*, pp. 39–53, Springer, 2000.

33. R. A. Murphey, “An approximate algorithm for a weapon target assignment stochastic program,” in *Approximation and Complexity in Numerical Optimization*, pp. 406–421, Springer, 2000.

34. D. K. Ahner and C. R. Parson, “Optimal multi-stage allocation of weapons to targets using adaptive dynamic programming,” *Optimization Letters*, pp. 1–13, 2014.

35. P. A. Hosein, M. Athans, *et al.*, *Preferential Defense Strategies.* Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, 1990.

36. B. Xin, J. Chen, Z. Peng, L. Dou, and J. Zhang, “An efficient rule-based constructive heuristic to solve dynamic weapon-target assignment problem,” *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 41, no. 3, pp. 598–606, 2011.

37. L. Wu, H. Wang, F. Lu, and P. Jia, “An anytime algorithm based on modified GA for dynamic weapon-target allocation problem,” in *IEEE Congress on Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence)*, pp. 2020–2025, IEEE, 2008.

38. D. Khosla, “Hybrid genetic approach for the dynamic weapon-target allocation problem,” in *Aerospace/Defense Sensing, Simulation, and Controls*, pp. 244–259, International Society for Optics and Photonics, 2001.

39. D. P. Lötter and J. H. Van Vuuren, "Weapon assignment decision support in a surface-based air defence environment," *Military Operations Research, Submitted*, 2014.

40. P. A. Hosein, J. T. Walton, and M. Athans, "Dynamic weapon-target assignment problems with vulnerable C2 nodes," tech. rep., Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, 1988.

41. R. M. Soland, "Optimal terminal defense tactics when several sequential engagements are possible," *Operations Research*, vol. 35, no. 4, pp. 537–542, 1987.

42. C. Leboucher, S. Le Menec, A. Kotenkoff, H.-S. Shin, and A. Tsourdos, "Optimal weapon target assignment based on an geometric approach," in *Automatic Control in Aerospace*, vol. 19, pp. 341–346, 2013.

43. O. Karasakal, "Air defense missile-target allocation models for a naval task group," *Computers & Operations Research*, vol. 35, no. 6, pp. 1759–1770, 2008.

44. D. P. Bertsekas, M. L. Homer, D. A. Logan, S. D. Patek, and N. R. Sandell, "Missile defense and interceptor allocation by neuro-dynamic programming," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 30, no. 1, pp. 42–51, 2000.

45. U.S. Joint Chiefs of Staff, *Joint Publication 3-01: Countering Air and Missile Threats*, 2014.

46. W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, vol. 703. John Wiley & Sons, 2011.

47. B. Van Roy, D. P. Bertsekas, Y. Lee, and J. N. Tsitsiklis, "A neuro-dynamic programming approach to retailer inventory management," in *Proceedings of the*

36th IEEE Conference on Decision and Control, vol. 4, pp. 4052–4057, IEEE, 1997.

48. W. B. Powell and B. Van Roy, "Approximate dynamic programming for high dimensional resource allocation problems," *Handbook of Learning and Approximate Dynamic Programming*, pp. 261–280, 2004.

49. M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, 2013.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
24–03–2016	Master's Thesis	SEP 2014 — MAR 2016

4. TITLE AND SUBTITLE	
Determination of Fire Control Policies via Approximate Dynamic Programming	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)	
Davis, Michael T., Captain, USAF	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433-7765	AFIT-ENS-MS-16-M-100

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
Intentionally Left Blank	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution Statement A.
Approved for Public Release; distribution unlimited.

13. SUPPLEMENTARY NOTES

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

Given the ubiquitous nature of offensive and defensive missile systems, the catastrophe-causing potential they represent, and the limited resources available to countries for missile defense, optimizing the response to a missile attack is a necessary endeavor. For a single salvo of offensive missiles launched at a set of targets, a missile defense system must decide how many interceptors to fire at each missile. Since such missile engagements often involve the firing of more than one attack salvo, we develop a Markov decision process (MDP) model to examine the optimal fire control policy for the defender. Due to the computational intractability of using exact methods for all but the smallest instances, we utilize an approximate dynamic programming (ADP) approach to explore the efficacy of applying approximate methods. We obtain policy insights by analyzing subsets of the state space that reflect a range of defender interceptor inventories. Testing of four scenarios demonstrates that the ADP policy provides high-quality decisions for a majority of the state space, achieving a 7.74% mean optimality gap. Moreover, computational effort for the ADP algorithm requires only a few minutes versus 12 hours for the exact DP algorithm, providing a method to address more complex and realistically-sized instances.

15. SUBJECT TERMS

missile defense, weapon-target, Markov decision process, approximate dynamic programming

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Lt Col Matthew Robbins, AFIT/ENS
					19b. TELEPHONE NUMBER *(include area code)*
U	U	U	U	71	(937) 255-3636, x4539 matthew.robbins@afit.edu

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18